CRANFIELD UNIVERSITY

LUKE FEETHAM

ROBUST VISION BASED SLOPE ESTIMATION AND ROCKS DETECTION FOR AUTONOMOUS SPACE LANDERS

CRANFIELD DEFENCE AND SECURITY

PhD
Academic Year 2016 - 2017

Supervisor: Dr. Nabil Aouf
March 2017

CRANFIELD UNIVERSITY

CRANFIELD DEFENCE AND SECURITY

PhD Thesis

Academic Year 2016 - 2017

Robust Vision Based Slope Estimation and Rocks Detection for Autonomous Space Landers

Supervisor: Dr. Nabil Aouf
March 2017

This thesis is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

ABSTRACT

As future robotic surface exploration missions to other planets, moons and asteroids become more ambitious in their science goals, there is a rapidly growing need to significantly enhance the capabilities of entry, descent and landing technology such that landings can be carried out with pin-point accuracy at previously inaccessible sites of high scientific value. As a consequence of the extreme uncertainty in touch-down locations of current missions and the absence of any effective hazard detection and avoidance capabilities, mission designers must exercise extreme caution when selecting candidate landing sites. The entire landing uncertainty footprint must be placed completely within a region of relatively flat and hazard free terrain in order to minimise the risk of mission ending damage to the spacecraft at touchdown. Consequently, vast numbers of scientifically rich landing sites must be rejected in favour of safer alternatives that may not offer the same level of scientific opportunity. The majority of truly scientifically interesting locations on planetary surfaces are rarely found in such hazard free and easily accessible locations, and so goals have been set for a number of advanced capabilities of future entry, descent and landing technology. Key amongst these is the ability to reliably detect and safely avoid all mission critical surface hazards in the area surrounding a pre-selected landing location.

This thesis investigates techniques for the use of a single camera system as the primary sensor in the preliminary development of a hazard detection system that is capable of supporting pin-point landing operations for next generation robotic planetary landing craft. The requirements for such a system have been stated as the ability to detect slopes greater than 5 degrees and surface objects greater than 30cm in diameter.

The primary contribution in this thesis, aimed at achieving these goals, is the development of a feature-based,self-initialising, fully adaptive structure from motion (SFM) algorithm based on a robust square-root unscented Kalman filtering framework and the fusion of the resulting SFM scene structure estimates with a sophisticated shape from shading (SFS) algorithm that has the potential to produce very dense and highly accurate digital elevation models (DEMs) that possess sufficient resolution to achieve the sensing accuracy required by next generation landers. Such a system is capable of adapting to potential changes in the external noise environment that may result from intermittent and varying rocket motor thrust and/or sudden turbulence during descent, which may translate to variations in the vibrations experienced by the platform and introduce varying levels of motion blur that will affect the accuracy of image feature

tracking algorithms. Accurate scene structure estimates have been obtained using this system from both real and synthetic descent imagery, allowing for the production of accurate DEMs. While some further work would be required in order to produce DEMs that possess the resolution and accuracy needed to determine slopes and the presence of small objects such as rocks at the levels of accuracy required, this thesis presents a very strong foundation upon which to build and goes a long way towards developing a highly robust and accurate solution.

Keywords:

Structure From Motion, Shape From Shading, Hazard Detection, Digital Elevation Models, Pin-point Landing, Adaptive Filtering.

ACKNOWLEDGEMENTS

First and foremost, I would like to deeply thank my supervisor Dr. Nabil Aouf, who has demonstrated seemingly endless patience with me and the numerous difficulties encountered during the course of this project. I could not have hoped for a more understanding and competent supervisor. His guidance and advice during this endeavour has been invaluable and without which the completion of this thesis would not have been possible.

I would also like to thank the European Space Agency and Airbus Defence ans Space for funding this exciting project, and providing me with the opportunity to visit ESTEC in The Netherlands, where I was able to carry out some very valuable practical work and meet many interesting people working in the space industry. Special thanks also goes to my supervisors at these two organisations, Olivier Dubois-Matra and Clement Bourdarias, for their helpful advice during the course of this project.

Very special thanks is also due to my family. My Mother, Linda Feetham, Father, Mark Feetham, and Sister Hayley Feetham, whose love and support over the years has been invaluable in helping me get to where I am today. I would also like to deeply thank my beautiful Fiance, Gemma Lowbridge, for her love and patient support during the completion of this PhD and previous MSc, despite her wishing that I would have gotten “a real job” and more fairly paid my way. I honestly could not have done this without you, and so I am eternally grateful and I will love you always.

I would also like to thank Cranfield University and the Defence Academy for providing a great environment for research and giving me the opportunity to undertake this project and continue on into an academic career.

TABLE OF CONTENTS

LIST OF PUBLICATIONS

Journal Papers

Single Camera Absolute Motion Based Digital Elevation Mapping for a Next Generation Planetary Lander

L. M. Feetham, N. Aouf, C. Bourdarias, and T. Voirin, “Single Camera Absolute Motion Based Digital Elevation Mapping for a Next Generation Planetary Lander”, *Acta Astronaut.*, vol 98, pp 169-188, May 2014.

A Self Initialising and Fully Adaptive Recursive Structure From Motion Algorithm for Hazard Detection Systems in Future Entry, Descent and Landing

In preparation

A Combination of Structure From Motion and Shape From Shading for Hazard Detection in Pin-Point Autonomous Planetary Landing

In preparation

Conference Papers

Single Camera Absolute Structure from Motion using H∞ Data Fusion for a Next Generation Planetary Lander

L. Feetham, N. Aouf, C. Bourdarias, and T. Voirin, “Single Camera Absolute Structure from Motion using H-infinity Data Fusion for a Next Generation Planetary Lander”, presented at the 5th European Conference for Aeronautics and Space Sciences (EUCASS), Munich, Germany, 2013.

Self Tuning, Recursive Structure From Motion for Hazard Detection in Autonomous Planetary Descent

L. M. Feetham, N. Aouf, C. Bourdarias, T. Voirin, and O. Dubois-Matra, “Self Tuning,

Recursive Structure From Motion for Hazard Detection in Autonomous Planetary Descent", presented at the 9th International ESA Conference on Guidance, Navigation and Control Systems (GNC 2014), Porto, Portugal, 2014

Image Datasets for Autonomous Planetary Landing Algorithm Development

L. Feetham, N. Aouf, O. Dubois-Matra, and C. Bourdarias, "Image Datasets for Autonomous Planetary Landing Algorithm Development", presented at the 7th International Conference on Mechanical and Aerospace Engineering (ICMAE 2016), London, United Kingdom, 2016

Combined Shape from Shading and Structure From Motion for Hazard Detection in Autonomous Planetary Landing

Under review

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

LIST OF ACRONYMS

ALHAT Autonomous Landing and Hazard Avoidance Technology

CCD Charge-Coupled Device

CMOS Complementary Metal-Oxide-Semiconductor

DEM Digital Elevation Model

EDL Entry Decent and Landing

EH$_\infty$F Extended H$_\infty$ Filter

EKF Extended Kalman Filter

ESA European Space Agency

FOV Field Of View

IMU Inertial Measurement Unit

IMU-KLT Inertial Measurement Unit assisted Kanade-Lucas-Tomasi feature tracker

KF Kalman Filter

KLT Kanade-Lucas-Tomasi feature tracker

LIDAR LIght Detection And Ranging

MS-SRUKF Master-Slave Square-Root Unscented Kalman Filter

NASA National Aeronautics and Space Administration

NEAR Near-Earth Asteroid Rendezvous

PANGU Planet and Asteroid Natural scene Generation Utility

PSO Particle Swarm optimisation

SFM Structure From Motion

SFS Shape From Shading

SIFT Scale Invariant Feature Transform

SIRFS Shape Illumination and Reflectance From Shading

SR-UKF Square-Root Unscented Kalman Filter

TR-KLT Time Reversible KLT

UKF Unscented Kalman Filter

1 INTRODUCTION

One of the most important, and yet one of the most risky operations of all planetary surface exploration missions is the entry, descent and landing (EDL) phase. This process is responsible for delivering the spacecraft to its intended landing site, during which the spacecraft must be decelerated from around 6–7km/s (for Mars missions) at atmospheric entry to near-zero velocity for a safe touchdown. EDL typically lasts for around 15 minutes and involves a sequence of critical tasks that must be executed reliably and with precise timing. It is often the case that, due to the large distances involved, delays in radio communication between the spacecraft and mission control are of the order of the duration of the entire EDL phase or longer (e.g. 13 minutes one-way communication delay between Earth and Mars during EDL of the Curiosity Rover [1]). Therefore, it is often impossible to carry out real-time manual control of the spacecraft during EDL, leading to the requirement for EDL systems to operate autonomously. This chapter discusses the current state of the art of EDL systems, the hardware that is most often employed and its limitations. Following this the need for greater landing precision and more sophisticated autonomy is presented, and a review of the computer vision-based techniques that have been proposed to achieve this is given. Finally, we conclude with a discussion of the main contributions of this thesis.

1.1 Current Entry, Descent and Landing Systems

To date, all entry descent and landing (EDL) systems for landing spacecraft on remote planetary surfaces have been predominantly based on technologies and concepts developed during the early days of the space age. For U.S. missions in particular, these systems are mostly based on those used for the Viking Mars landers, which landed on the surface of Mars in 1976 [2] . All missions since have used this technology as the backbone design, and incorporated only slight modifications aimed at increasing landing accuracy and reliability. However, even these modifications are often based on previous technology; therefore there is a limit to how much the system can be improved. For example, the 1997 Mars Path Finder mission employed the entry and descent technology from the Viking landers and incorporated terminal landing architecture derived from the 1971 Soviet Mars 2 & 3 landers [2] as well as an airbag system that can be traced back to the Soviet Luna 9 Moon lander of 1966 [3].

The EDL systems that have been in use from the start of the space age up to the present day are known as first generation EDL systems [4]. Such systems typically have large landing uncertainty ellipses, with major axis lengths ranging from 200–300km (Pathfinder, Viking) down to around 80km (Mars Exploration Rovers) [5], and typically result in a hard landing (velocity = 10–50m/s) [4]. With this type of system, mission designers are often forced to trade mission safety and scientific interest, and choose landing zones such that the majority of the landing ellipse falls within regions of relatively flat and safe terrain to maximise the chance that the spacecraft will survive the landing [5, 6]. This approach therefore places tight restrictions on the choice of landing site.

The reason for such large landing uncertainties in previous missions is due to the methods used for navigation before and during entry and descent. Currently, space landers are navigated with respect to initial position knowledge of the spacecraft prior to entry interface, which may contain significant errors as this determination is often carried out through Earth-based observations [7]. This initial position is then propagated solely using on-board inertial sensors throughout the entry phase and some of the descent phase, which leads to a significant degradation in accuracy as a consequence of error accumulation. During the rest of the descent

phase, the height of the lander is often determined using a radar altimeter, which can also enable vertical velocity determination. Horizontal velocity is also often measured using Doppler radar. These radar systems enable velocity control to ensure a safe touchdown, and can be used to reduce further error accumulation in the inertial sensors, but cannot be used to adequately correct for the horizontal position error already accumulated during the inertial navigation phase. Additionally, this suite of sensors is unable to provide any information on surface hazards. Thus, current EDL systems are incapable of providing a safe, pin-point landing at a specific location on a planetary surface.

In future missions, with their ever increasing focus on reduced cost and risk as well as greater scientific return, the need for a precise pin-point landing at a site of significant scientific interest, or in close proximity to previously deployed assets, is becoming a highly coveted capability for next generation EDL. In general, scientifically interesting landing sites are not flat and often contain many landing hazards, including significant variation in terrain elevation, craters, and rocks [5]. Thus, for a spacecraft to land at a specific location on such terrain, it must possess a number of sophisticated capabilities. These capabilities include: precisely identifying the intended landing site and determining position with respect to the landing site; accurately estimating motion parameters in all 3 dimensions; reliably detecting hazards such as highly sloped terrain, rocks, crater rims, cliff edges, etc. and be able to identify alternative safe landing sites in the event that the original landing site is determined to be unsafe; finally, it should be capable of precisely controlling its trajectory in all six degrees of freedom in order to steer towards the intended landing site and avoid any detected hazards.

Standards have been set regarding advanced capabilities of EDL systems for future NASA missions as part of a project named Autonomous Landing and Hazard Avoidance Technology (ALHAT). This project has specified that future space landers should be capable of detecting surface hazards larger than 0.3m, slopes greater than 5 degrees, and have the ability to navigate to a safe landing site with a diameter of 15m within 100m of a previously specified location [8–10]. Although the ALHAT project is primarily focused on autonomous landing on the moon in support of aspirations of future robotic and manned NASA missions, the goals for landing precision and hazard avoidance are applicable for a wide range of planetary exploration missions launched by any of the world's space agencies (e.g. these goals are also of particular interest to the European Space Agency (ESA)).

A number of concepts have been proposed in order to achieve such precision and reliability for landing on planetary surfaces. The two main ones, especially when it comes to hazard detection, include the use of either computer vision based systems or LIDAR based systems [11]. LIDAR, which is an acronym for Light Detection And Ranging, uses the same principle as radar except that it emits pulses of laser light and detects the light reflected back from the surface in order to measure distance. Scanning LIDAR systems would be able to generate very accurate and dense digital elevation models (DEMs) of the surface during descent that could be used to reliably identify surface hazards. However, LIDAR scanners typically have a mass of around 10kg, consume around 25W of power [4], require significant on-board processing power, and are a relatively unproven technology for space applications, making them an expensive choice of instrument purely for use in EDL. Cameras, on the other hand, have a long history of use in space; they are low cost, low power, low mass devices and are carried by almost all spacecraft for scientific imaging and optical navigation. Thus their use in autonomous navigation for the purpose of landing on a planetary surface can be realised with little or no impact on typical spacecraft systems [12].

As a consequence of the benefits of using cameras as measurement instruments mentioned above, computer vision techniques have been widely pursued for the development of next generation EDL. Research in this area has mostly focused on two main autonomous capabilities: absolute position and motion estimation, and hazard detection and avoidance. The techniques proposed for absolute position determination tend to rely on having prior information (obtained

from orbital measurements) on the 3D location of certain landmarks present at the intended landing site and attempt to match features observed in the decent imagery with the known features in orbital images. Thus the spacecraft must possess orbital images/terrain models/landmark databases in on-board storage for reference during the descent. The form of reference data used can have a significant impact on the design and performance of the system. Examples of these techniques can be found in [7, 13–17]. Research into hazard detection has made use of a variety of techniques for identifying and characterizing hazards such as rocks, craters, steep slopes and surface discontinuities. Examples of some of the techniques that have been investigated include: rock detection by shadows [18–20], jim texture analysis for determining surface roughness and discontinuities [8, 20], shape from shading for determining surface slope [20], and homography slope estimation [8, 20].

1.2 Project Aims

The aims of this PhD project are to develop innovative concepts and techniques for accurately estimating slopes and detecting other surface hazards such as small rocks that may present potentially significant risks to the health of the spacecraft at touchdown. These capabilities are to be provided via the use of on-board visual sensors and through the implementation of robust computer vision algorithms and robust filtering techniques to enable accurate and reliable estimates of the required terrain features. In addition to this, it is desirable that accurate and dense metric estimates of terrain elevation are obtained to provide sufficiently detailed information for a sophisticated hazard detection and avoidance module. Indeed, although rocks can be detected using shadows, it is likely that dense 3D terrain modelling will be a prerequisite for truly robust small rocks detection, in addition to being complimentary in estimating surface slopes. Therefore, significant emphasis will be placed on constructing an accurate digital elevation model (DEM) of the terrain around the intended landing site.

1.3 Review of Proposed Vision-Based Techniques

The first spacecraft to successfully land on a planetary body using vision based navigation techniques was the Near-Earth Asteroid Rendezvous (NEAR) spacecraft, on 14th February 2001. What was remarkable about this accomplishment was that the spacecraft was not designed to achieve this feat, yet the sophisticated optical navigation system was able to provide the necessary accuracy in the trajectory for the spacecraft to make a safe touchdown. The landing was sufficiently gentle for the spacecraft to sustain no damage and to remain operational despite not having any form of landing gear [21].

The optical navigation system of the NEAR-Shoemaker spacecraft was designed to facilitate orbit determination during the terminal approach and orbit phase of the mission, and was the first mission to make operational use of landmarks identified in images. The adopted method consisted of locating easily identifiable craters on the surface of Eros-433 and using those landmarks to infer the relative position and thus trajectory of the spacecraft, as well as to infer the rotation state of the asteroid, and the body-fixed coordinates of each landmark. Each landmark was identified manually by the navigation team, and measured by dragging a cursor around the rim of the crater. The (x, y) coordinates of the centre of each crater would then be computed in software by fitting an ellipse to the set of points marked by the operator and then calculating the central point of the resulting best-fit ellipse. Although labour intensive, this approach was found to produce accurate results for the spacecraft trajectory, generally providing an uncertainty of around 10m in the spacecrafts orbit [22]. The same technique was applied to the landing scenario, and was found through post-landing analysis to have enabled a vertical

velocity of 1.5–1.8m/s (target = 1.3m/s) and a transverse velocity of 0.2–0.3m/s (target = 0.2m/s), and enabled the lander to touchdown within 500m of the target landing site [23].

Despite the success of the optical navigation system of NEAR-Shoemaker and the unanticipated proof of concept that it provided through the initially unplanned landing on Eros, the manual processing of landmarks was an onerous and time consuming activity – accurate orbit determination was calculated on average twice a week during the orbit phase, and during the descent phase 3 landmarks were identified and used in 4 out of 12 images after 15 minutes of receipt of the images [22]. Indeed, had the landing been a critical part of the mission from the outset, and had the mission not been in the low gravity environment of a comet, allowing for extremely low velocities and hence time to allow for manual intervention, it is doubtful that this technique would have been employed. The landing site was selected for its scientific interest, but mostly to minimise risk to the spacecraft (smooth terrain and scarcity of boulders) and because it was less sensitive to orbit timing errors due to the shape of the gravitational potential [21]. Thus the landing site was selected based more on opportunity than anything else.

For missions in which it is critical that a spacecraft lands at a specific point of scientific interest, a much greater level of accuracy and control would be required. Additionally, there may be significant light-time delays between the spacecraft and Earth (e.g. 4–24 minutes between Earth and Mars [1]) and the velocity of the spacecraft is likely to be far in excess of those of Near-Shoemaker during its orbit phase, which was generally only a few m/s [21]. Therefore it is clear that the spacecraft must possess significant autonomy, since arduous and time consuming manual processing would be out of the question.

Even though the NEAR mission was the first operational demonstration of the use of vision in descent and landing navigation, the use of these methods had been under investigation for some time prior to NEAR-Shoemaker. Cuseo et al [24] investigated the use of machine vision techniques for the purpose of autonomously guiding an unmanned spacecraft to a safe landing on Mars. Their work concentrated on 3 areas: (1) monocular techniques to detect rough surface hazards such as boulder fields; (2) optical flow or shape from motion techniques to determine the 3D structure of the terrain and detect hazards such as steep slopes; (3) recognition of landmarks to reduce navigation errors and achieve a more precise landing, as well as assisting in hazard avoidance. For rough surface hazards they employed a simple technique involving identifying pixels that deviate significantly from their nearest neighbours, which was motivated by the need to detect objects of 1m size that would be sub-pixel for a portion of the descent. This technique would lead to the identification of regions of sufficient size and smoothness for a safe landing. The optical flow technique consisted of tracking edge points in the image and using the displacement of these points and the known camera motion to determine the range to the points. Kalman filtering is used to incrementally improve the range estimates and to provide an estimate of the uncertainty in each range. A surface is then interpolated to the tracked edge points to produce a map that can be used to identify hazardous regions such as steeply sloped terrain. For landmark tracking they employed a correlation method to identify surface features from stored templates of features that had been identified from prior orbital imaging of the landing site. The features were matched to the templates using a generalised Hough transform and evaluating multiple combinations of scale, rotation and viewing perspective based on the estimated trajectory parameters. The image location of the correlation and the template identifier (whose coordinates are known) are then used to update the navigation state of the lander. Results from a real-time hardware simulation using scaled terrain models indicated that landing errors could be reduced from 1km to 30m using these techniques.

Poelzleitner and Paar [25] studied a scenario that focussed on landing site identification and position initialisation during the early stages of descent, using pre-stored images and digital elevation models of the landing site, which had been pre-selected and analysed using orbital imagery. Computer vision methods were applied to images captured by a single camera on-board the lander in order to match the camera view with the stored images of the landing site,

thus enabling the position of the spacecraft to be determined with respect to the planned landing spot. In their scenario, a 3D digital elevation model was generated using stereo imagery of the landing site recorded by an orbiting spacecraft. These images were to be transmitted to Earth for processing in order to generate the 3D digital elevation model as well as an image of the entire landing ellipse, in which a number of landmarks had been identified and their coordinates computed. The image and elevation model were then transmitted back to the spacecraft and stored on-board the landing craft for use during descent. A robust matching algorithm, based on feature vector matching, was applied to identify the portion of the landing ellipse that was in the FOV of the camera in the first captured frame, and to identify features corresponding to the pre-computed landmarks in the overall landing ellipse image. Once a number of features had been matched, the on-board digital elevation model can be used to determine the 3D coordinates of the features and thus compute the absolute position of the spacecraft. In this work, the authors paid particular attention to the specific altitude at which the lander should begin capturing and processing the images. The purpose of this was to ensure that the resolution of the images captured by the lander camera was the same as that of the images recorded from orbit, in order that the image scales match between the orbiter and lander. This highlights an important consideration that must be taken into account when trying to identify a pre-selected landing site, even though a number of more modern feature matching algorithms have been demonstrated to be robust under changes of scale. The authors also suggested that the images taken by the orbiter segment may have to be manipulated on Earth in order to produce images that would have the same viewing angle and illumination conditions as those expected from the lander during descent to facilitate comparison during the matching process. Again, this must also be taken into consideration, despite there being a number of more modern techniques available that incorporate rotation, affine transformation, and illumination invariance measures.

Paar and Polzleitner [26] applied the techniques developed in [25] to a Moon landing scenario. The scenario was tested using a scale model to simulate the spacecrafts descent onto the lunar surface, and it was found that navigation errors of less than 3m could be achieved from this method.

Johnson and Matthies [27] developed an algorithm for on-board motion estimation to enable precision guidance for autonomous landing on small bodies using a monocular vision system. Their technique falls into the category of two-frame feature-based motion estimation, and thus consists of automatic feature detection and tracking between a pair of descent images, followed by 2-frame motion estimation, which is then coupled with laser altimetry data for scale recovery. For each pair of images (recorded simultaneously with a pair of laser altimetry readings) a number of features are identified at random locations in the 1st image using the feature detection algorithm of Benedetti and Perona [28], which is itself a variation of the Shi and Tomasi [29] feature detector and tracker. Since their application is intended for small body landing, for which descent velocities are likely to be small and the motion is likely to be relatively free of sudden changes, the detected features are tracked in the 2nd image using the optical flow based method of Shi and Tomasi [29]. The tracked features are then fed into a linear algorithm, which is applied multiple times using different sets of features to eliminate outliers and obtain a consistent set of features, and then to obtain a robust least median of squares estimate of the motion. This motion estimate is used as an initial estimate in a more accurate nonlinear algorithm that solves for the motion parameters directly. The output of this algorithm is an estimate of 5 motion parameters along with their covariance, which can then be combined with the laser altimetry readings to recover the scale and solve for the 6th motion parameter. Two different techniques were proposed for how to use the altimetry data based on the surface topography of the landing site (smooth or rugged terrain) and the characteristics of the descent trajectory (mostly vertical motion or with significant horizontal motion). The system was tested using real imagery obtained from a gantry test bed and with simulated altimetry readings and was found to be of high accuracy. The most accurate results for a simulated landing were obtained for a pure vertical

descent, which was calculated to result in a landing position accuracy of 3.6m over a 1000m descent, i.e. a 0.36% motion error. Motion errors were also calculated for pure horizontal motion and for a $45°$ descent, which resulted in 1.8% and 1.3% errors respectively. Rotational motion errors of 1% were also obtained. Although the results are very promising, the system would need to be flight tested to determine its performance in a real-life situation.

Roumeliotis et al [30] continued the work of Johnson and Matthies [27] by incorporating inertial sensor data to refine the motion estimates produced by the vision algorithm. Some modifications were made to the vision algorithm to make it more applicable for landing on a planetary surface. In this situation, significant shifts in rotation and translation are likely to occur during the descent phase. To handle this, the feature tracking method was changed to a correlation based technique instead of an optical flow based approach. The correlation method adopted was that described in [31]. The motion estimation obtained from the images is similar to [27] except that the initialisation for the non-linear filter comes from a simple global search of a coarse discretisation of the rotation and translation space to find the minimum of a cost function, instead of using a robust 8-point algorithm. Other than this the method is the same. The processing of the relative pose measurements from the image based motion estimation and the fusion of the IMU measurements is carried out using an indirect Kalman filter. The system was tested on a gantry test bed, and although no estimated final landing site errors were stated by the authors, significant improvements can be seen in their results when compared to vision or IMU alone.

Bajracharya [32] focussed on real-time methods for hazard detection from a single image taken during the descent. The real-time constraint meant that only simple methods could be employed due to the complexity of image processing algorithms and limitations of computing hardware at the time. The surface is assumed to change slowly in intensity when there are no hazards present, therefore hazardous areas are identified when sudden changes of intensity are detected. The image is first segmented by examining local intensity variations via a k-means clustering algorithm that clusters pixels together based on their texture features. Outliers from this clustering algorithm are then classified as either hazards or as shadows. If a region is identified as a shadow, it is further examined to determine the potential hazard that may have cast the shadow, such as rocks or craters. Once the entire image has been analysed a hazard map is produced that highlights safe and unsafe areas. A similar approach to this is described in [33], which also uses a clustering algorithm to detect rocks, but it does this on multiple scales via the use of an image pyramid in order to detect rocks of varying size. Such an approach could be a useful means of the rocks detection needed in this project. Other techniques that focus on hazard detection are: [18] that uses shadows to identify rocks from images; [19] that uses edge detection and ellipse fitting to identify craters, intensity gradients and homography-based slope estimation to identify steep slopes and discontinuities such as cliff edges, and shadow analysis to identify rocks; [20] that uses stereo vision for slope estimation and rocks detection, as well as shadow analysis for additional rocks detection; and [8] that uses corner and edge detection coupled with intensity variance to identify highly textured regions of an image, which are likely to be caused by the presence of rocks, steep slopes, craters etc. Hoff and Sklair [34] adopt a different strategy for hazard detection. They employ a feature tracking algorithm to track edge points across an image sequence and use a separate Kalman filter for each feature point to estimate the 3D location of the feature in order to construct an interpolated surface that represents the landing site terrain. Using this estimated surface, hazards such as steep slopes can be identified.

The majority of methods in the literature that aim to employ visual sensors to perform a pin-point landing focus on the motion estimation and navigation aspect of the problem. This has mostly been achieved through the use a priori knowledge about the landing site such as previously mapped landmarks measured from orbit. These known landmarks are then identified in descent images using feature matching methods either in the 2D images themselves or in on-board estimated 3D surface models that are compared with preloaded on-board digital

elevation models from orbital surveys. By using these known landmarks the absolute position and motion of the spacecraft can be determined, allowing a safe, pin-point landing at the intended landing site. Examples of this type of approach can be found in [7, 13, 14, 17, 35–37]. Other approaches have used only information extracted from the images during descent in order to carry out terrain relative navigation, such as in [14, 38].

1.4 Contributions of Thesis

The primary focus of this thesis is on the development of preliminary methodologies for enabling autonomous hazard detection capabilities that are sufficient to support pin-point landing operations with future planetary lander spacecraft. The adopted approach for achieving this is to use a single on-board descent camera for the estimation of 3D terrain structure using powerful and sophisticated modern computer vision techniques. As will be discussed in subsequent chapters, the problem of estimating scene structure from camera imagery is closely coupled with the problem of estimating the motion of the camera and so these are often estimated simultaneously (using methods known as structure from motion (SFM) or simultaneous localisation and mapping (SLAM)). Consequently, this thesis also presents the development and testing of a robust structure from motion method with respect to its ability to accurately estimate the motion parameters of the spacecraft as well as the scene structure, despite the focus being on hazard detection. Since depth information is lost in the projection of a 3D scene onto a 2D image plane, the use of a single camera introduces problems in determining the scale of objects observed in the images. Consequently, motion and structure can only be recovered up to an unknown scale factor in the absence of additional information. This thesis describes a method of fusing measurements from an IMU in order to directly estimate the unknown scale to enable fully scaled metric estimates of both the scene structure and the motion parameters. Therefore, unlike the other methods described in the literature and mentioned above, the method developed here is able to provide accurate and robust motion and structure estimates without the need of any prior 3D knowledge of the terrain surrounding the landing site.

A spacecraft descending towards a planetary surface can be subject to many complex dynamic disturbances and influences, such as atmospheric turbulence, sudden shocks from the firing of retro-rocket motors, variable rocket thrust during powered descent, etc. Each of these will result in varying levels of vibration, which will manifest as varying quantities of image blur that will affect the accuracy of measurements derived from descent images in a complex and unpredictable fashion. This realisation motivated the further development the above mentioned structure from motion algorithm into a fully adaptive and self-initialising algorithm based upon two square-root unscented Kalman filters operating in a parallel master-slave configuration. In this context, the master filter is responsible for estimating the scene structure and the slave filter is responsible for adaptively re-tuning the master filter in the event of changing noise statistics. This is the first time such a filtering framework has made use of the square-root unscented Kalman filter. The two filters were also optimally initialised using particle swarm optimisation, and also incorporated more traditional adaptive filtering methodologies in order to adaptively re-tune the slave filter, if necessary, which further increases the novelty of the approach. Following on from this, a sophisticated method of using shading information to recover scene structure was examined and applied to descent imagery. This was then combined with the previous structure from motion results in order to merge the two highly complimentary techniques to produce dense 3D digital elevation models of the terrain surrounding the landing site. To this author's knowledge, this is the first time such a combination of the two techniques has been carried out within the context of hazard detection in EDL.

2 CAMERA MODELS AND MEASUREMENTS FROM IMAGES

This chapter discusses the concepts and techniques involved in the use of digital camera systems as measurement devices and the types of measurements that are commonly made from images in computer vision. It begins with a discussion of camera models and the geometry and mathematics commonly used to describe the image formation process of a 2D image on the image plane of a camera from the 3D scene within the camera's field of view. This project makes use of a monocular imaging system, i.e. a single camera, as the primary means of taking exteroceptive measurements of the local environment. Therefore this chapter will primarily describe the concepts behind monocular imagery along with its strengths and limitations, however, a brief discussion of stereo camera systems will also be given in order to provide justification for why this type of camera system has not been exploited in this project.

2.1 Camera Models

In order for cameras to be used as measurement devices, it is necessary to understand how 3D scene points map onto the 2D image plane of the camera. The most common way of examining this is with a simple geometric model known as the pinhole camera model, as described in detail in Subsection 2.1.2. This assumes that all light rays enter the camera through a tiny pinhole aperture such that only a single ray from any given point in the scene will enter the camera and form an image on the image plane [39]. This feature of the pinhole camera is what enables sharp images to be produced on the image plane - light rays emanating from any particular point in the scene may be reflected from light coming from many different directions, but since a pinhole camera only lets a single light ray through then this will result in a very sharply focussed image [39]. However, such a model is an ideal case and as such is difficult to achieve in practice. Also, by only letting individual light rays enter the camera, long exposure times are required for the image sensor to be able to register a legible image [39], which can result in such camera systems being impractical in the majority of situations. In real camera systems, a lens is used, which, due to the refraction of light rays through the lens material, enables more light to be collected and focussed onto the image plane, allowing for shorter exposure times and brighter images whilst also allowing a sharp image to be produced. We therefore begin with a brief discussion of a simple model of lenses before coming back to the pinhole camera model. Later, in Section 2.2, we describe a more sophisticated model of the effects that lenses can have on the image, in terms of lens distortions and how these effects can be corrected mathematically, but for now we stick to rather more simplistic and idealised mathematical descriptions of image formation.

2.1.1 Thin Lenses

To reduce the exposure time required to form a legible image, the light collecting capability of the camera must be increased. This necessarily means that a larger number of rays emanating from each individual scene point must be allowed to enter the camera aperture and fall upon the image plane. One way of achieving this would be to increase the diameter of the camera aperture - i.e. the pinhole. However, each scene point will reflect a collection of rays, perhaps from multiple light sources, or due to scattering, reflections from other objects, the non-parallel nature of the

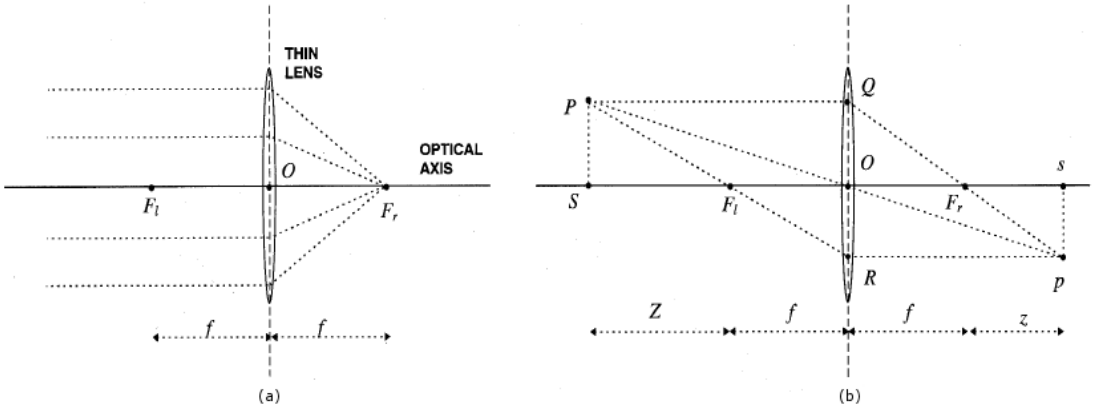


Figure 2.1: Geometric properties of thin lenses: (a) all incident parallel rays are focussed to a single point; (b) image formation of a point in the scene - F_l and F_r are the left and right focal points, respectively, f is the focal length, and O is the lens centre (image source: [39])

rays emanating from the light source, or the light source having a not-insignificant physical size. Each of these reflected rays will therefore have a slightly different angle of reflection, producing a cone of light rays with vertex at the scene point and spreading further apart as distance from the scene point increases. The end result of this is that each scene point becomes slightly spread out across the image plane and therefore a reduction in the sharpness of the image is observed. To overcome this problem, an optical system consisting of lenses, apertures and other elements is introduced that is explicitly designed to refract incoming light rays so that all rays emanating from a single scene point will be refracted such that they converge to a single point on the image plane [39].

The basic function of any optical system, however sophisticated, can be understood using the concept of an ideal thin lens, the geometric behaviour of which is illustrated in Figure 2.1. The fundamental optical properties of the thin lens are shown in Figure 2.1, and are summarised as follows [39]:

- Any ray entering the lens parallel to the optical axis on one side goes through the focus on the other side.
- Any ray entering the lens from the focus on one side emerges parallel to the optical axis on the other side.

Figure 2.1 shows the way in which the light rays emanating from the scene point P undergo refraction as they pass through the lens and form an image p of P on the image plane. The point at which the image comes into focus can be identified by intersecting only two known rays [39], which we can construct using the rules listed above, i.e. by drawing a ray emanating from P that is parallel to the optical axis, and one that passes through the focus F_l in front of the lens. These two rays will then intersect at the point p on the other side of the lens. However, we can also draw another ray that intersects the lens at the point O, which will pass through the lens undeflected and also intersect with the other two rays at p. Using these three rays and the similar triangles method, we can derive an equation that describes the location of the image point for any given scene point. This equation is known as the fundamental equation of thin lenses, and is expressed as:

$$\frac{1}{\hat{Z}} + \frac{1}{\hat{z}} = \frac{1}{f} \tag{2.1}$$

where $\hat{Z} = Z + f$ and $\hat{z} = z + f$. Figure 2.1 (b) could be extended to include all the possible light rays emanating from the scene point P, which would show that all of these rays would intersect at the point p on the right hand side, producing a sharply focussed image. The net effect is that the camera's light gathering power is increased through the use of a lens, but otherwise it is behaving in much the same way as a pinhole camera – the ray that passes through the point O is the same ray as that which would be the only ray admitted by the pinhole camera and any other rays are refracted by the lens and focussed onto the same image point. The contribution of the lens is therefore to reduce the length of time that the shutter would need to be open in order for the image sensor to sufficiently register the image. We can therefore proceed with our analysis of how scene points map to image points by considering the simple geometry of pinhole cameras.

2.1.2 Pinhole and Perspective Camera Models

A pinhole hole camera consists of a light-proof box with an image plane on one side, upon which an image is projected by the light rays that enter through a tiny aperture (as if made by piercing the box with a pin) on the opposite side. Unlike with a lens-based camera, the light rays entering the aperture of a pinhole camera undergo no distortion or refraction, thus they follow a straight line path from the scene point to the image point. Figure 2.2 illustrates the geometry of the image formation process for a pinhole camera, which can be used to derive the following equations relating the 3D scene points to the 2D image points through the application of similar triangles:

$$\begin{aligned} x' &= f\frac{X}{Z} \\ y' &= f\frac{Y}{Z} \end{aligned} \quad . \tag{2.2}$$

Notice the relative orientation of the two coordinate axes on the right of Figure 2.2. If the primed coordinate frame, centred on the principal point, had the same orientation of the unprimed coordinate frame, centred on the centre of projection (the pinhole), then Equation 2.2 would be re-written as

$$\begin{aligned} x' &= -f\frac{X}{Z} \\ y' &= -f\frac{Y}{Z} \end{aligned} \quad ,$$

which is an illustration of the fact that the image produced by a pinhole camera is inverted, as can be seen in the left side of Figure 2.2.

The image inversion issue can be avoided by treating the image plane as if it were in front on the centre of projection and considering it to be a transparent plane with the observer at the centre of projection. The image is therefore considered to be produced by the points at which the light rays emanating from the objects in the scene intersect with the transparent image plane. This arrangement is illustrated in Figure 2.3. When represented in this form it is known as the perspective camera model, which may be regarded as the default camera model within computer vision. With this model the image plane coordinate frame, centred on the principal point, has the same orientation as the camera reference frame coordinate axes, centred on the centre of projection.

From Figure 2.3 it can be seen that the perspective camera model consists of the image plane sitting at a distance f (the focal length) in front of the centre of projection, O. The line through

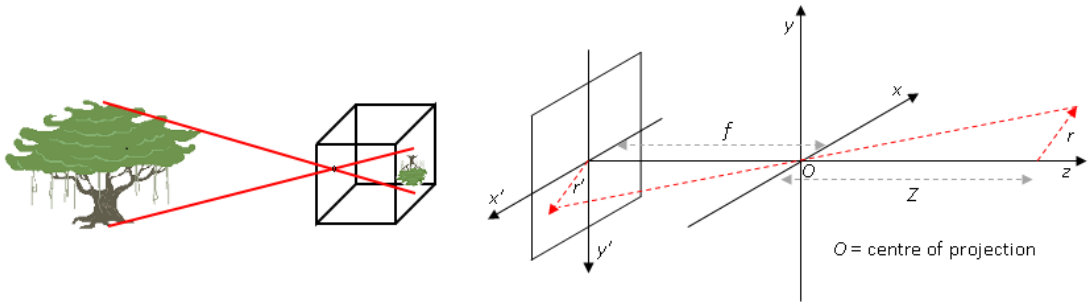


Figure 2.2: Image formation and geometry of the pinhole camera model. Note: the symbols have the same meaning as in Figure 2.1 and in Equation 2.1

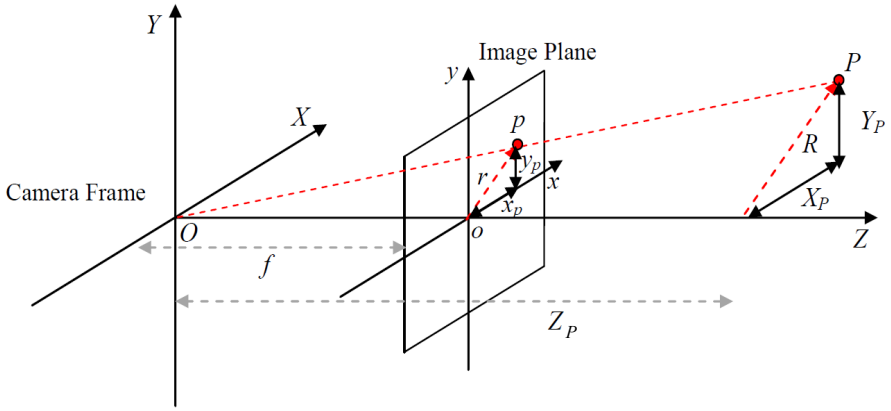


Figure 2.3: Geometry of the perspective camera model

O and perpendicular to the image plane is the optical axis, and o, the intersection between the image plane and the optical axis, is known as the principal point or image centre. The image of point P on the image plane is denoted p and lies on the straight line through P and O at the point where this line intersects the image plane. By considering a 3D reference frame where O is the origin and the image plane is perpendicular to the Z axis, the point P is represented by the vector $P = [X_P, Y_P, Z_P]^T$. This reference frame is known as the camera reference frame. If the coordinates of the point p in the image plane are given by x_p and y_p, the point p can be expressed in vector form as $p = [x_p, y_p, f]^T$, where the z_p coordinate is the distance of the image away from the centre of projection, i.e. $z_p = f$ [39].

Using similar triangles, the x_p and y_p coordinates of the image point p can be expressed in terms of the scene point coordinates X_P, Y_P and Z_P in the camera frame, therefore enabling us to link the position of scene points with that of their corresponding image points [39]. This leads to the basic equations of perspective projection, which are expressed as:

$$x_p = f\frac{X_P}{Z_P} \quad , \quad y_p = f\frac{Y_P}{Z_P} \tag{2.3}$$

or in vector form as

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \frac{f}{Z_P} \begin{bmatrix} X_P \\ Y_P \end{bmatrix}. \tag{2.4}$$

Note that Equation 2.3 is the same as Equation 2.2, except that now we are dealing with non-inverted coordinate axes and an image that is also not inverted. This is a much more intuitive way of visualising the relationship between a scene and its image.

Digital images are captured by an image sensor such as a CCD or a CMOS active pixel sensor. These devices consist of a rectangular grid of photo-sensors that convert the incoming light rays into a voltage that is proportional to the incoming light intensity. These voltages are read out from the image sensor, and then digitised into a 2D array or matrix of integer values. The elements of this 2D matrix are known as pixels, where each pixel represents the light intensity captured by an element of the photo-sensor (assuming a one-to-one correspondence between pixels and photo-sensor elements).

Since computer vision algorithms work with digital images, the only information available to the algorithms regarding the location of objects of interest in the scene are the pixel coordinates of the image points of these objects in the digital image. The equations for the x and y coordinates of an image point in relation to the 3D scene point (Equation 2.4), are written in the camera reference frame, and represent the physical location of the image point in relation to this frame. Therefore, in order to proceed, a relationship between the pixel coordinates and the image point coordinates in the camera frame is required.

Relating the pixel coordinates to the camera frame requires knowledge of the characteristics of the camera. These characteristics are known as the intrinsic and extrinsic parameters. The extrinsic parameters are the parameters that define the location and orientation of the camera reference frame with respect to some known world reference frame. The intrinsic parameters are the parameters that link the pixel coordinates of an image point with the corresponding coordinates in the camera reference frame [39].

2.2 Intrinsic and Extrinsic Camera Parameters

2.2.1 Extrinsic Parameters

The camera frame is fundamental to the perspective camera model and to the equations that are derived from it that describe the relationship between image points and the corresponding scene points. However, the camera frame is often unknown [39], and thus introduces a problem when trying to relate the image information to the structure of the 3D world that the images represent. What is needed is some way of determining the location and orientation of the camera frame with respect to a known world frame using only image information [39]. This is the purpose of the extrinsic parameters. For example, this is precisely what is done for the VISILAB images in Chapter 6, in which images of a calibration pattern are used to measure the distance of the camera from the terrain surface.

The extrinsic parameters are defined as any set of geometric parameters that uniquely identify the transformation between the unknown camera reference frame and a known reference frame, often called the world reference frame [39]. Typically these geometric parameters take the form of a 3D translation vector, $\mathbf{T}$, describing the relative displacement of the origins of the two reference frames, and a 3x3 rotation matrix, $\boldsymbol{R}$, which is an orthogonal matrix that can be thought of as describing the rotation that would be required to bring the corresponding axes of the two frames into alignment with each other [39]. Therefore, the relationship between the coordinates of a point in the world frame, $\mathbf{P}_w = [X_w, Y_w, Z_w]^T$, to those of the same point in the camera frame,

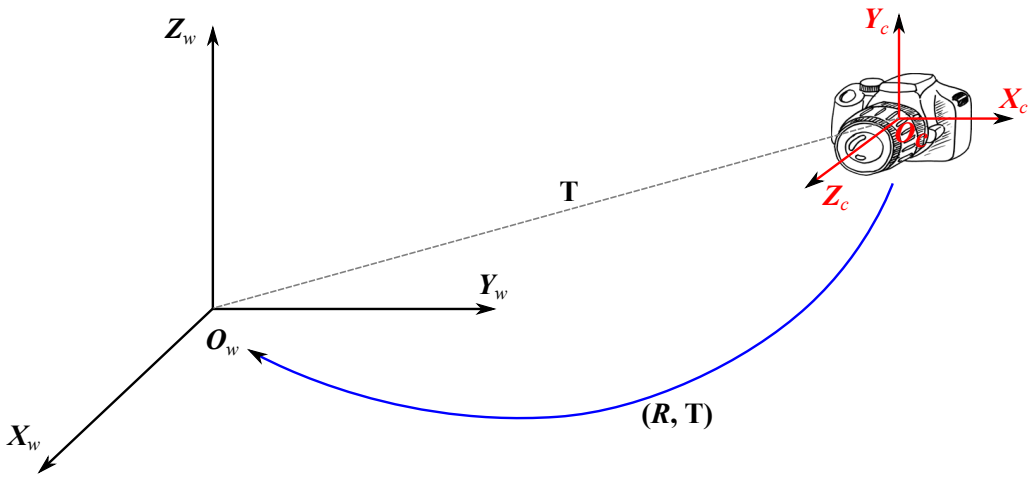


Figure 2.4: Transformation between the potentially unknown camera reference frame and the known world frame coordinate systems

$\mathbf{P}_c = [X_c, Y_c, Z_c]^T$, can be expressed in two ways:

$$\mathbf{P}_c = \boldsymbol{R}(\mathbf{P}_w - \mathbf{T}_w) \tag{2.5}$$

or

$$\mathbf{P}_c = \boldsymbol{R}\mathbf{P}_w + \mathbf{T}_c \tag{2.6}$$

The first of these two equations describes a translation followed by a rotation, whereas the second equation describes a rotation followed by a translation. The net effect is the same, however, it is important to note that while the rotation matrices will be the same in either case, the translation vectors will be different [39]: in Equation 2.5 the translation vector represents the displacement of the origin of the camera frame with respect to the origin of the world frame, and is expressed in the world frame coordinate system, hence it is expressed with the subscript w; in Equation 2.6 the translation vector represents the displacement of the origin of the world frame with respect to the origin of the camera frame, and is expressed in the camera frame coordinate system, hence it is expressed with the subscript c. This process of translation and rotation between the two frames is illustrated in Figure 2.4.

The extrinsic parameters of the camera therefore consist of the 3 elements of the 3x1 translation vector $\mathbf{T}$, and the 9 elements of the 3x3 rotation matrix $\boldsymbol{R}$. However, since $\boldsymbol{R}$ is an orthogonal matrix describing a valid 3D rotation, thus it has a determinant of $+1$ and obeys the following relationship:

$$\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I},$$

therefore the number of free elements of $\boldsymbol{R}$ reduces to 3. Thus, in total, there are 6 extrinsic parameters that need to be determined for each recorded image in order to uniquely describe the position and orientation of the camera reference frame with respect to the world frame during the capture of the particular image in question.

2.2.2 Intrinsic Parameters

The intrinsic parameters can be defined as the set of parameters needed to characterise the optical, geometric, and digital characteristics of the camera, allowing the pixel coordinates of an image point to be related to the coordinates of that point in the camera reference frame [39]. This requires three sets of parameters:

1. The focal length, f, which specifies the perspective projection
2. Parameters specifying the transformation between camera frame coordinates and pixel coordinates
3. Parameters specifying any geometric distortion introduced by the optics

Due to difficulties in manufacturing, it is unlikely that the centre of the image will coincide with the principal point in the camera reference frame. Therefore, in order to express the camera frame coordinates in terms of the pixel coordinates, we must determine the displacement of the image point from the optical axis.

The coordinates of points of interest in the image are given in terms of pixel coordinates, i.e. in units of pixels, with reference to the 2D image pixel coordinate system with origin at the top left corner of the image, whereas the image coordinates in the camera frame represent real physical coordinates measured with respect to the origin of the camera reference frame or equivalently (if only considering the 2D location of image points on the image plane) with respect to the principal point. Therefore, assuming a one-to-one correspondence between image pixels and photo sensor elements, to convert the pixel coordinates into camera frame coordinates we must multiply by the image sensor element size and translate with respect to the coordinates of the principal point. The resulting equations for relating image pixel coordinates to camera frame coordinates are therefore expressed as:

$$\begin{aligned} x_c &= -\left(x_{im} - o_x\right) s_x \\ y_c &= -\left(y_{im} - o_x\right) s_y \end{aligned} \quad , \tag{2.7}$$

where $(x_c,\, y_c)$ are the 2D coordinates of the image point on the image plane with respect to the camera reference frame, $(x_{im},\, y_{im})$ are the coordinates of the image point in units of pixels measured with respect to the top-left corner of the image, $(o_x,\, o_y)$ are the coordinates of the principal point in units of pixels measured with respect to the top-left corner of the image, and $(s_x,\, s_y)$ are the physical sizes of the image sensor elements [39]. The negative signs in these two equations are due to the fact that the axes in the image pixel coordinate frame are in the opposite directions to those of the camera reference frame as illustrated in Figure 2.5. Note that it is possible to define the camera reference frame with a different orientation in order to avoid this difference in axes direction and consider only a translational shift of the origins of the image pixel frame and the camera frame. This can slightly simplify the equations by avoiding the leading minus signs in Equation (2.7). Providing one is consistent in all calculations then this should not introduce any problems. This is what was done in the structure from motion algorithms presented later in this thesis.

The above equations (Equation (2.7)) are not the full story. In fact they are only valid for the pinhole camera model, or for cameras with optics that conform perfectly to the ideal thin lens model. In reality, lenses do not conform to the ideal model leading to geometric distortions and most likely there will be additional distortions introduced due imperfections in the lens and misalignments in the construction of the camera due to less than perfect manufacturing techniques. These distortions effectively cause the pixels to be in a different location in the image than what would be expected, resulting in phenomena such as the “barrel” or “fish-eye” effect [40].

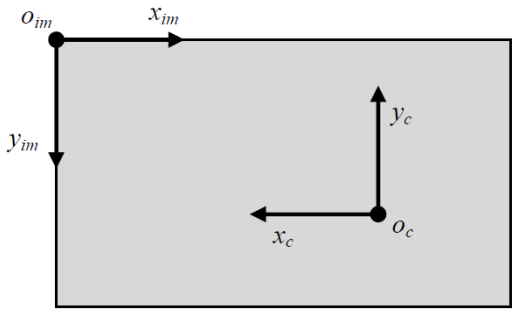


Figure 2.5: Relative orientations of the coordinate axes of the image pixel coordinate frame and the camera reference frame. Note that the origin of the image pixel coordinate axes is at the top left corner of the image and that the principal point is not necessarily at the centre of the image.

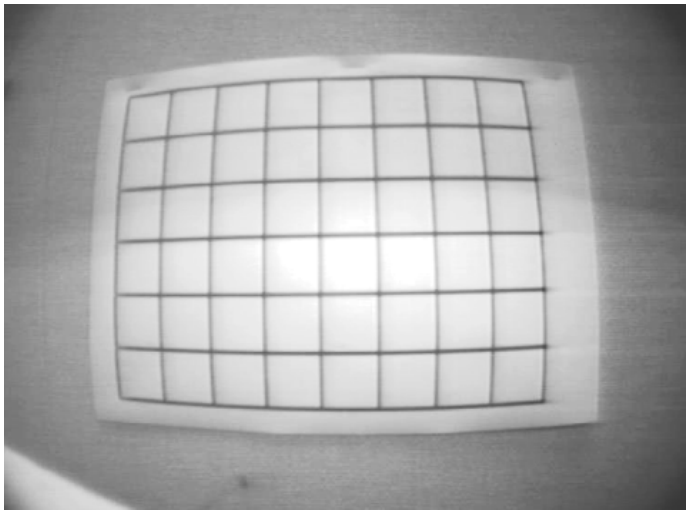


Figure 2.6: Image showing the effects of radial distortion (image source: [41])

Lens distortions arise due to irregularities in the shape of the lens and due to the fact that it is easier to manufacture lenses that are spherical rather than parabolic in their overall shape [40]. The dominant effect of these properties of real lenses is to introduce radial distortions, where the magnitude of the distortion increases radially from the principal point of the image. These distortions are usually small and can be modelled accurately using the first few terms of a Taylor expansion around radius $r = 0$ [40], which results in equations of the form:

$$\begin{aligned} x_{im} &= x_d \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \\ y_{im} &= y_d \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \end{aligned} \quad , \tag{2.8}$$

where $(x_d,\, y_d)$ are the pixel coordinates of the distorted points and $r^2 = x_d^2 + y_d^2$. These equations show that the distortion increases as r increases and that at $r = 0$, there is zero distortion [39]. For the majority of cameras, only the first 2 terms are normally required the third term is usually only necessary for very wide angle lenses such as fish-eye lenses [39, 40]. The effect of radial distortion can be seen in Figure 2.6, which shows that points that are further from the optical axis, e.g. the outside corners of the grid pattern, are displaced towards the centre by a greater amount

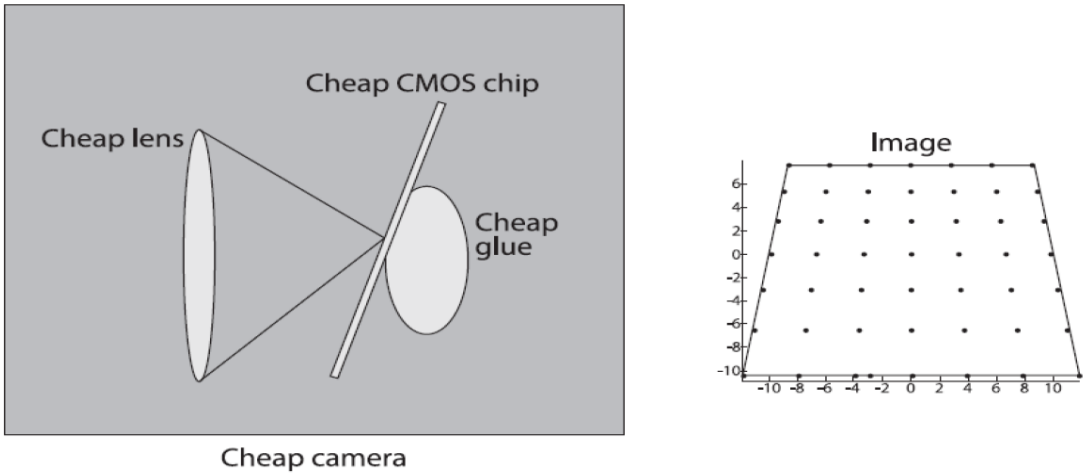


Figure 2.7: Cause and effect of tangential distortions (image source: [40])

than points that are closer to the optical axis. This is what causes the barrel effect seen in the image.

The other significant type of distortion that may be present in an image is known as tangential distortion, which arises due to misalignments of the image plane with respect to the lens and vice versa [40]. These are principally due to manufacturing defects and/or the use of cheap materials, but may also arise through mistreatment of the camera over time. Tangential distortion is characterised by two parameters p_1 and p_2 such that [40]:

$$\begin{aligned} x_{im} &= x_d + \left\{2p_1 y_d + p_2\left(r^2 + 2x_d^2\right)\right\} \\ y_{im} &= y_d + \left\{p_1\left(r^2 + 2y_d^2\right) + 2p_2 x_d\right\} \end{aligned} \quad , \tag{2.9}$$

where the symbols have the same meaning as in Equation (2.8). The cause and effect of this type of distortion are illustrated in Figure 2.7.

To summarise, therefore, the complete list of intrinsic parameters are the focal length f, the pixel coordinates of the principal point $(o_x,\, o_y)$, the dimensions of the image sensor elements $(s_x,\, s_y)$, the radial distortion parameters k_1 and k_2 (ignoring k_3), and the tangential distortion parameters p_1 and p_2, giving 9 parameters in total.

2.3 Camera Calibration

In order to reliably make use of measurements obtained from images in computer vision algorithms the values of the intrinsic parameters need to be estimated to a high degree of accuracy. Once these parameters have been fully characterised, any distortions that may be present in the images can be corrected for by applying suitable transformations to the images. This operation is known as undistortion and results in images that can be treated in algorithms as if they were recorded by a camera that obeys the simple image formation geometry of the pinhole or perspective camera. The process that is employed to estimate the intrinsic parameters is known as camera calibration and involves capturing multiple images of a planar object from a wide variety of different viewing angles, orientations and distances from the camera. This planar object contains a regular pattern of easily detectable points with known geometry and by detecting the precise locations of these points in all images of the set of

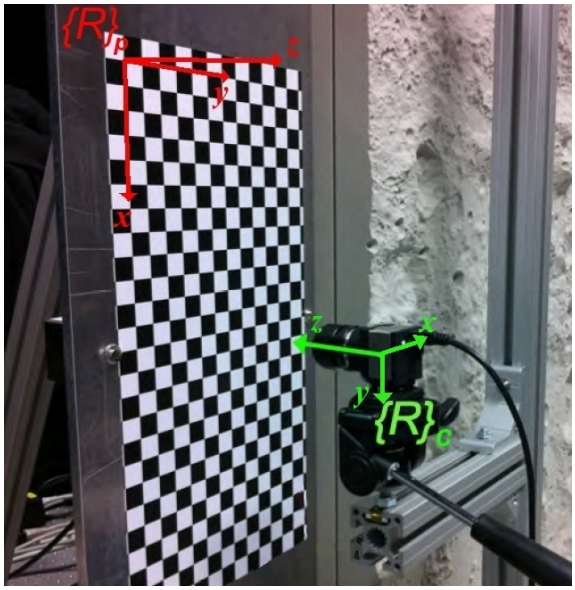


Figure 2.8: Camera calibration pattern used in this project for calibrating the VISILAB camera: consists of 24x 12 (usable) black and white squares of size 15x15mm

calibration images, and, using the known geometry, allows the intrinsic parameters of the camera to be estimated. The most common type of pattern used for camera calibration is a chessboard type pattern of black and white squares of a constant and known physical size. Each image of the calibration pattern also has its own set of 6 extrinsic parameters describing the relative displacement and orientation of the camera reference frame and the calibration pattern reference frame (the world frame in this case) during the capture of that particular image, thus the extrinsic parameters for each image are also estimated during calibration as these are required in solving for the intrinsic parameters. The specific calibration pattern used in camera calibration in this project is shown in Figure 2.8 and has a square size of 15x15mm. Further information on the specific practical procedure used to calibrate the camera in this project will be given in Chapter 6, whereas here we shall focus on the theoretical aspects of camera calibration.

In order to present the mathematical details of camera calibration, we first need to re-write the previous equations in a slightly different form, so that we may combine them. This is achieved by making use of a concept known as planar homography, which describes the perspective transformation of a planar object when it is viewed through a pinhole or lens and projected onto another planar object (the image plane), thus planar homography is defined as a planar mapping from one plane to another [40].

Dropping the p subscripts from Equation (2.3) and combining this equation with Equations (2.6) and (2.7), we can write

$$\begin{aligned} -\left(x_{im}-o_x\right)s_x &= f\frac{\mathbf{R}_1^T\mathbf{P}_w+T_x}{\mathbf{R}_3^T\mathbf{P}_w+T_z} \\ -\left(y_{im}-o_y\right)s_y &= f\frac{\mathbf{R}_2^T\mathbf{P}_w+T_y}{\mathbf{R}_3^T\mathbf{P}_w+T_z} \end{aligned}, \tag{2.10}$$

giving an expression that relates the image pixel coordinates to the world frame coordinates without explicit reference to the camera frame, where $\mathbf{R}_i$, $i=1,2,3$, is a 3-element column

vector formed from the i^{th} row of the rotation matrix $\boldsymbol{R}$ [39]. This equation can be re-written in terms of a matrix product by defining two matrices $\boldsymbol{M}_{int}$ and $\boldsymbol{M}_{ext}$, as

$$\boldsymbol{M}_{int} = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \quad , \tag{2.11}$$

$$\boldsymbol{M}_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix}$$

so that $\boldsymbol{M}_{int}$ depend only on the intrinsic parameters, and $\boldsymbol{M}_{ext}$ depends only on the extrinsic parameters [39]. Now, by adding a "1" as a forth coordinate of $\mathbf{P}_w$, i.e. expressing $\mathbf{P}_w$ in homogeneous coordinates[1], Equation (2.10) can be expressed equivalently as:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \boldsymbol{M}_{int}\boldsymbol{M}_{ext} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad , \tag{2.12}$$

where

$$\begin{aligned} x_1/x_3 &= x_{im} \\ x_2/x_3 &= y_{im} \end{aligned} \quad . \tag{2.13}$$

Equation (2.10) in the form of Equation (2.12) is now in the form

$$\tilde{\mathbf{q}} = s\boldsymbol{H}\tilde{\mathbf{Q}} \quad , \tag{2.14}$$

where

$$\begin{aligned} \tilde{\mathbf{q}} &= \begin{bmatrix} x_{im} & y_{im} & 1 \end{bmatrix}^T \\ \tilde{\mathbf{Q}} &= \begin{bmatrix} X_w & Y_w & Z_w & 1 \end{bmatrix}^T \end{aligned}$$

and x_3 has been factored out and taken across to the other side and expressed as s to denote that it is an arbitrary scale factor [40]. The matrix $\boldsymbol{H} = \boldsymbol{M}_{int}\boldsymbol{M}_{ext}$ is known as the homography matrix, and contains all of the information about the camera that we are trying to determine.

An important simplification can be made to the homography matrix by considering the fact that the object coordinates all lie on the same plane (the planar calibration pattern). Thus we can set the object coordinate Z_w to zero without any loss of generality to give $\tilde{\mathbf{Q}}'$, which is defined

[1] Given a point (x, y) on the Euclidean plane, for any non-zero real number Z, the triple (xZ, yZ, Z) is called a set of homogeneous coordinates for the point. The original Cartesian coordinates for a point (xZ, yZ, Z) can be recovered by dividing the first two elements by the third, e.g.

$$\begin{bmatrix} x/1 \\ y/1 \\ 1/1 \end{bmatrix} = \begin{bmatrix} fX/Z \\ fY/Z \\ Z/Z \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} fX/Z \\ fY/Z \\ 1 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z}\begin{bmatrix} X \\ Y \end{bmatrix}.$$

only for the plane we are interested in, instead of $\tilde{\mathbf{Q}}$, which is defined for all of space. In doing this one of the columns of $\boldsymbol{R}$ then becomes unnecessary [40]:

$$\begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = s\boldsymbol{M}_{int} \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} = s\boldsymbol{M}_{int} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \quad . \tag{2.15}$$

Therefore H is completely defined by the scale factor s, the intrinsics matrix $\boldsymbol{M}_{int}$, the translation vector $\mathbf{T}$, and the first two columns of the rotation matrix $\boldsymbol{R}$ [40].

It is Equation (2.15) that is used to solve for the camera parameters during camera calibration, by computing the homography matrix for multiple views of the calibration pattern. Since a rotational transformation is described by a set of three rotation angles, and a translation is described by three offsets, each view of the calibration pattern results in six unknowns for the extrinsic parameters plus the four unknowns for the intrinsic parameters (which are the same in each view). This may seem like a problem because with every view we are adding an extra six unknowns that must be calculated from the image data, however, each view of the calibration pattern gives 8 equations, which allows 8 unknowns to be calculated [40]. Thus the full set of intrinsic parameters could be obtained from just two images of the calibration pattern. However, by using a large number of images the parameters can be estimated to a much higher accuracy through the use of optimisation methods.

The preceding equation is initially used to compute the intrinsic parameters assuming that no distortion is present in the images. In reality, however, the object points will be in the wrong place in the images since this assumption is not valid. By combining Equations (2.8) and (2.9), the undistorted image points, here denoted $(x_p,\, y_p)$, are related to the distorted image points, $(x_d,\, y_d)$ (which are $(x_{im},\, y_{im})$ in Equation (2.15)), by

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \begin{bmatrix} x_d \\ y_d \end{bmatrix} + \begin{bmatrix} 2p_1 y_d + p_2\left(r^2 + 2x_d^2\right) \\ p1\left(r^2 + 2y_d^2\right) + 2p_2 x_d \end{bmatrix} \quad . \tag{2.16}$$

The use of multiple images allows these extra intrinsic parameters to be estimated by collecting a large list of these equations and solving for the distortion parameters. After these parameters have been estimated the previous intrinsics parameters are re-estimated, and the whole process is repeated until the desired accuracy is achieved [40].

2.4 Stereopsis

The above discussion dealt with camera models and calibration with respect to a single camera. In this section we discuss the principles behind stereo camera systems, which allow for a direct determination of the 3D location of scene points via triangulation. The principles developed above are still applicable, except now we have two cameras to deal with, which introduces some additional concepts.

A typical stereo camera rig consists of two cameras side by side, with a relatively short baseline separation, an example of which is shown in Figure 2.9. For most purposes it is desirable for the two cameras to be perfectly aligned with each other, i.e. for their optical axes to be perfectly parallel. However, due to difficulties in manufacturing, this would generally be impossible to achieve. It is therefore common for the configuration to be similar to that shown in Figure 2.10, although generally to a lesser extent than what is indicated in this exaggerated schematic. This misalignment can cause problems when calculating distances using the stereo camera, thus this problem must be overcome before meaningful results can be obtained.

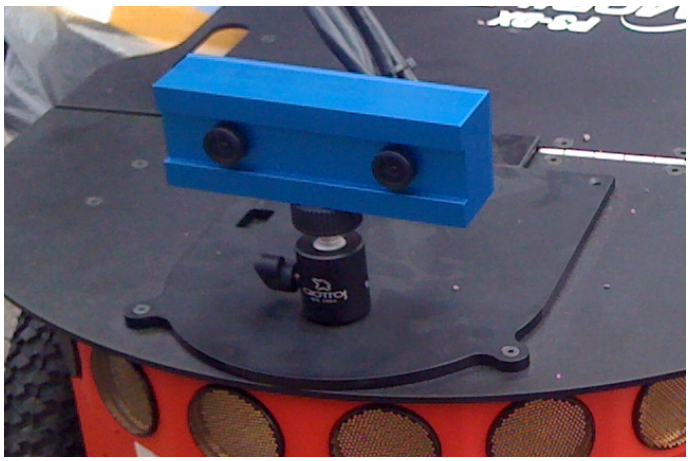


Figure 2.9: Example of a typical stereo camera

Another challenge that must be tackled when using stereo cameras is the problem of identifying corresponding features in the two images that are obtained in each frame. Additionally, the pixel locations of these corresponding features must not be distorted and must be known with high accuracy. Therefore the use of stereo cameras involves four key steps [40]:

1. Mathematically remove radial and tangential distortions from the two images by applying Equation (2.16) to each image using the distortion parameters estimated during calibration. This is known as undistortion, and the output of this step is a pair of undistorted images that are consistent with the simple perspective camera model.
2. Adjust for the misalignment between the cameras using the baseline separation parameter and rotation matrix describing the relative orientation of the two cameras (additional parameters that are obtained during stereo calibration). This process is known as stereo rectification and results in a pair of images that are row-aligned and rectified, which means that the resulting images are consistent with those that would be produced by a stereo system with perfectly coplanar image planes with the sensor elements being exactly row-aligned (i.e. parallel optical axes and equal pixel y-coordinates).
3. Find matching features between the left and right images. This is a process known as correspondence, and is carried out by exploiting the epipolar geometry (see [40] for details) of the stereo rectified images – matching features will have equal y-coordinates.
4. Compute the distances (Z_c-coordinate) for the corresponding features in the images through a process known as triangulation, using the idealised stereo camera geometry.

For further details on steps 1, 2, & 3 see [40]. We now assume that these steps have been carried out and move on to discuss triangulation (step 4) based on the simplified geometry of a perfectly aligned camera pair.

2.4.1 Triangulation

In step 4 of the above list, we have a set of corresponding features identified in each image of the stereo pair for which we want to compute the distance. To do this we need to make the use of the idealised stereo camera geometry that is obtained mathematically from stereo calibration (see [40] for details) and a method known as triangulation. Therefore, we assume that we have a stereo rig that consists of two identical cameras (i.e. with the same focal length) that are perfectly

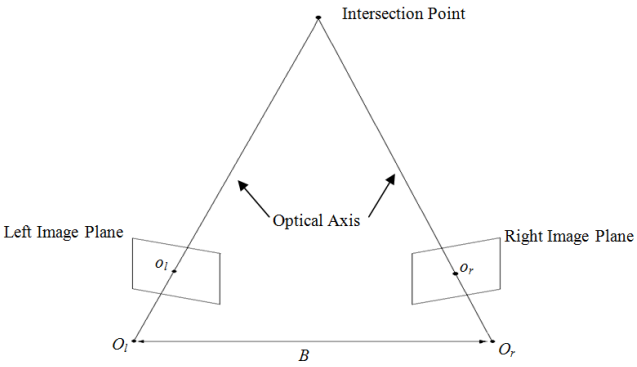


Figure 2.10: Exaggerated schematic showing the likely physical characteristics of stereo camera rigs, i.e. it will almost certainly exhibit some degree of misalignment (non-parallel optical axes) between the two cameras

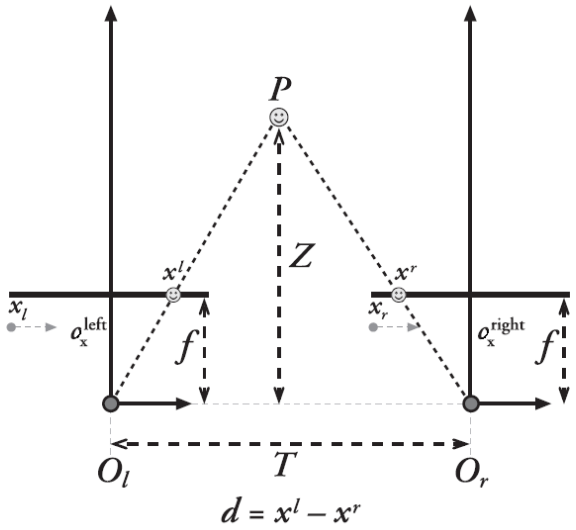


Figure 2.11: Ideal geometry of a stereo camera rig (image source: [40])

aligned and produce undistorted images. We also assume that the principal points of the two cameras, $\frac{l}{x}$ and o_x^r have been calibrated to have the same pixel coordinates in their respective left and right images [40]. The geometry of this set-up is then as shown in Figure 2.11.

A point P in the physical world will produce an image point p_l in the left image and an image point p_r in the right image. If x^l and x^r are the horizontal positions of the image points in the left and right camera frames respectively, and the cameras are separated by a known baseline T , the distance Z to the scene point can be derived using Figure 2.11 by considering similar triangles [40]:

$$\frac{T - \left(x^l - x^r\right)}{Z - f} = \frac{T}{Z}$$

therefore

$$Z = \frac{fT}{x^l - x^r} = \frac{fT}{d} \tag{2.17}$$

where $d = x^l - x^r$ is the disparity between the two view of the scene point P, which can be seen to be inversely proportional to the distance [40]. Note that in Figure 2.11 x^r is to the left of the optical axis and is therefore a negative quantity in the Figure.

Equation 2.17 reveals an important issue with the use of stereo cameras. Since depth is inversely proportional to the disparity, objects that are far away will have a disparity that is near zero, and a small change in the disparity will produce a large change in the calculated depth. However, when objects are close the disparity is large and small changes to the disparity will not change the depth by a significant amount. Consequently, stereo vision systems have high depth resolution only for objects that are relatively close to the camera [40]. This will therefore mean that for the purposes of its use in Kalman filter-based Simultaneous Localisation and Mapping (SLAM) or similar algorithms, objects that are far away will have a large measurement uncertainty and this uncertainty will not be linear with respect to depth. In other words, slight errors in the measured disparity due to influences such as image noise, feature tracking errors, etc. can introduce potentially very significant errors in the determination of the depth.

In this project we are considering a landing craft descending towards the surface and trying to construct an accurate 3D model of the terrain upon which the craft will touchdown in order to reliably detect the presence of potentially mission ending hazards. In this descent scenario, image based measurements of the terrain are expected to begin at an altitude of around 2000m above the surface. Taking, as an example, 2 uEye cameras with GOYO lens that is used in Chapter 6 as our pair of cameras, which have a 3.5mm focal length, and assuming a baseline separation of 20cm, at an altitude of 2000m the disparities of detected features in the image pair will be around $3.5\text{x}10^{-7}$m, which with a sensor element size of $5.3\text{x}10^{-6}$m is a disparity of approximately 0.07 pixels, and this will be similar for all detected features since the height variation of the terrain within the field of view of the camera is likely to be fairly small in comparison to the altitude. The localisation accuracy of sub-pixel interest point detectors is of the order of $0.1-0.02$ pixels [42], therefore, taking the best of these two uncertainties, the error in estimated depths from this altitude would be of the order of ± 600m, which is clearly unacceptable. For this reason, stereo camera systems are not considered as a viable means of estimating scene structure using descent imagery in this project. Instead, the use of a single camera (monocular imagery) is investigated.

2.5 Scale Ambiguity in Monocular Images

Due to the potentially very large errors in the estimated depths of scene points with a stereo camera system at the operating altitudes of this project it was decided early on to adopt a method based on monocular imagery, i.e. using a single camera. This approach leads to two main classes of technique for estimating the depths of image feature points from information contained within the interframe motion of these points from one image to the next: (1) the so called two-frame methods, in which the essential matrix relating two views of the scene is estimated from a collection of corresponding points (the essential matrix is a concept borrowed from stereopsis and therefore this method can suffer from similar problems as stereo vision systems when disparity is small - this can be overcome by increasing the baseline separation, which in this case is achieved by ensuring a large interframe displacement, e.g. by recording two images sufficiently separated in time) with estimates possibly being refined using batch processing techniques known as bundle adjustment; (2) or through some sort of filter-based state estimation technique, in which depth estimates are recursively refined with each successive image measurement of the feature point positions. Each of these classes of technique will be reviewed in detail in Chapter 4.

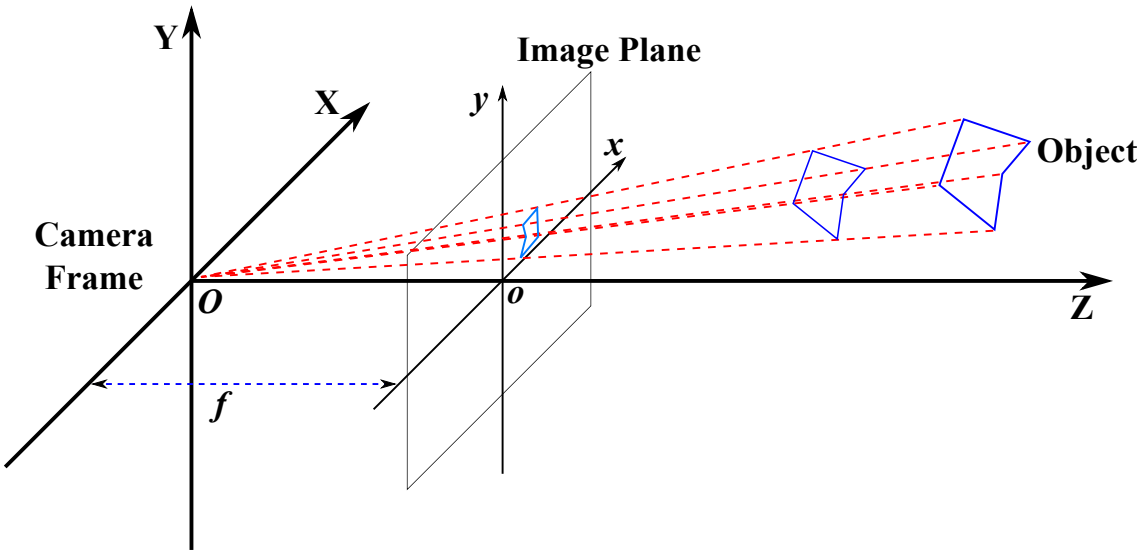


Figure 2.12: Illustration of the scale ambiguity problem in monocular imagery: A large far away object can produce an identical image to that of a smaller and closer object

The main problem with monocular scene reconstruction methods is that, with only a single camera, if the relative motion between the camera and the scene is unknown or the physical size of objects in the scene are unknown, then it is impossible to determine how far objects are from the camera - depth information is irreversibly lost in the projection from a 3D scene to a 2D image. The problems this introduces can be appreciated by examining Figure 2.12, which demonstrates how an identical image may be produced by a larger object far away from the camera and a smaller object closer to the camera. Therefore, vision based methods for estimating motion and/or recovering the 3D structure of the scene can, at best, only estimate such information up to an unknown scale factor in the absence of any additional information from outside sources.

Fortunately, landing craft, almost without fail, carry a number of sensors that can provide information relating to the true motion of the spacecraft. With the motion known, the scale ambiguity problem can be overcome by observing the changes in the image positions of the features over time – the way in which image features move from image to image will overwhelmingly (especially for far away objects) be due to the relative motion between the scene and the camera, but subtle differences from this primary image motion contains within it information relating to the depth of the features in the scene. The types of additional sensors that can provide information on true motion and that are often carried by spacecraft that are intended to land on planetary or other remote objects include such sensors as inertial measurement units, radar or laser altimeters, Doppler radar systems, etc. (see Chapter 1). With the use of measurements from these type of sensors a data fusion strategy can be derived that enables the direct estimation of a suitable scale factor that can be used to transform the unscaled estimates of scene structure and motion derived from the images into fully scaled metric quantities. An approach that makes use of inertial measurements for this purpose is presented in Chapter 4, in which is is demonstrated that very accurate estimates of motion and structure can be obtained even from high altitude for which the image motion is small. Later, it is assumed (under the advice of ESA) that the motion of the spacecraft is known (supplied by another on-board system) therefore the scale ambiguity issue is entirely avoided and there is no need to directly estimate a scale factor.

2.6 Conclusions

This chapter has described many of the details behind the use of digital camera systems as exteroceptive measurement devices. It has outlined the geometry of the pinhole and perspective camera models and described the concepts of camera calibration that together enable pixel-based image information to be related to metric quantities expressed in the camera coordinate frame of the camera system. A discussion on stereopsis was also given, in which it was described how a pair of cameras can be used to recover fully-scaled measurements of the depth of objects in the scene. However, example calculations demonstrated that at the initial height of the camera system above the surface of the planetary body, a stereo camera system would not be capable of providing the necessary measurement accuracy. Thus providing justification for rejecting stereo vision in favour of monocular imagery. Finally, the issues involved in using a monocular camera system, in terms of scale ambiguity, were outlined and methods of overcoming this ambiguity were briefly mentioned in advance of a full examination of this problem in Chapter 4.

3 FEATURE TRACKING

In monocular SFM, because of the scale ambiguity problem, and also because of the altitudes involved in this project, structure is a weakly observable property. Therefore, if a filter-based SFM algorithm is used, it is important that the image features are tracked for as long as possible in order to allow sufficient time for the structure estimates to converge to an adequate level of accuracy and hence allow for a proper decoupling of the structure from the motion. This chapter presents two different types of feature tracking methods: the Scale Invariant Feature Transform (SIFT), and the Kanade-Lucas-Tomasi (KLT) feature tracker. For the latter, two different variations of KLT are described and their relative performance is analysed in order to highlight the weaknesses inherent in the conventional formulation of KLT, and how these weaknesses can, to some extent, be overcome by a more robust formulation of KLT that is not so prone to the accumulation of errors that are typical in the conventional method. The main contributions of this chapter are a realisation that the SIFT feature matching method is likely unsuitable for robust 3D reconstruction in monocular SFM and that the KLT tracker should be implemented with additional constraints (e.g. the TR-KLT algorithm) in order to reduce error accumulation and maximise the number of images that features can be successfully tracked over. It is successfully demonstrated that the TR-KLT algorithm results in reduced error accumulation compared to conventional KLT.

3.1 Scale Invariant Feature Transform

The Scale Invariant Feature Transform (SIFT) is a feature detection and matching method first proposed by Lowe in 1999 [43] and later elaborated on in 2004 [44]. It is designed to identify image features that are invariant to image scale and rotation, and robust to a substantial range of affine distortion, 3D viewpoint change, image noise, and variation of illumination. The features detected using the SIFT method are highly distinctive and discriminative in the sense that an individual feature can be correctly matched with high probability against a large database of features from many images due to the utilisation of a highly unique and invariant image feature descriptor associated with each extracted interest point [43]. These powerful properties of the SIFT feature detector are particularly appealing for the use-case in this project because appearance changes due to scale will certainly occur in descent imagery as the camera gets closer to the surface.

The SIFT approach to extracting feature points (referred to as keypoints by Lowe) consists of 4 main steps: (1) scale-space extrema detection, (2) keypoint localisation, (3) orientation assignment, (4) keypoint descriptor calculation. These steps are summarised in the following subsections.

3.1.1 Scale-Space Extrema Detection

In this step, a search is conducted in all image locations over a wide range of scales to identify potential interest points that are invariant to image scale. This is achieved by first constructing a sampled scale-space from the original image. The construction of the scale space is similar in many respects to the construction of an image pyramid (as will be seen in the KLT algorithm below), where each level in the pyramid is an image a factor of 2 smaller in height and width than the image in the next level down, except that in the SIFT algorithm each pyramid level has a number of sub-levels that are of the same spatial size but have different scales with respect to the

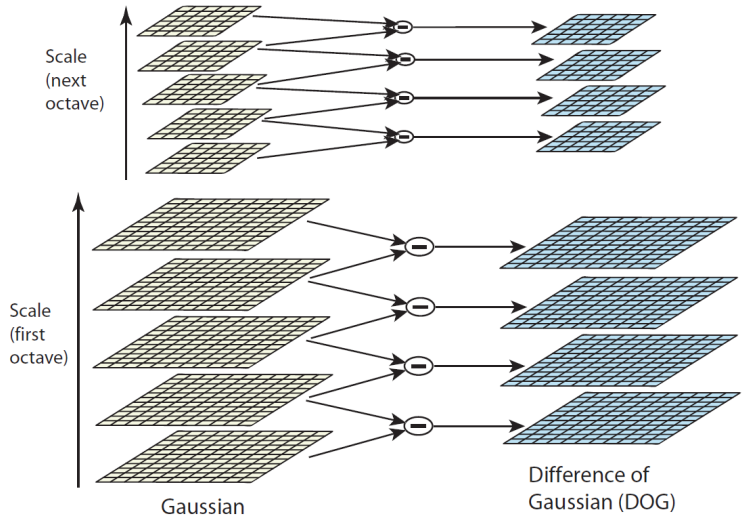


Figure 3.1: SIFT scale space pyramid (left), constructed by repeated convolution with a Gaussian kernel to obtain each sub-level image, followed by down-sampling to obtain each primary pyramid level. Adjacent sub-level images are subtracted to produce difference of Gaussian images (right). Image obtained from [44].

frequency domain. The sub-levels of each primary pyramid level are referred to as an octave, and are generated by repeated convolutions with a Gaussian kernel, with standard deviation σ. Thus, starting from the original image, a Gaussian blur operation is applied to the image to produce the next sub-level image, which is then convolved again with the same Gaussian kernel to produce the next sub-level image, and so on until the desired number of images in the octave is reached. The next primary level up in the pyramid is constructed by re-sampling the sub-level image that has twice the initial value of σ by taking every second pixel in each row and column. This next pyramid level is used to construct sub-level images by further Gaussian blur operations.

Following the construction of the scale-space, the actual interest points are identified by searching for scale-space extrema in difference of Gaussian images, which are obtained from the scale-space pyramid by subtracting adjacent scale images within each primary pyramid level – i.e. by subtracting adjacent sub-level images. This approach is illustrated in Figure 3.1. The identification of a scale-space extremum point is carried out by comparing each sample point (pixel) with its eight nearest neighbours in the current difference of Gaussian image and its nine nearest neighbours in the scale above and the scale below, as shown in Figure 3.2

3.1.2 Keypoint Localisation

The previous step identifies a set of candidate SIFT keypoints via neighbourhood pixel comparisons, with each keypoint being localised to pixel level accuracy. In this step, the candidate keypoint is localised to sub-pixel accuracy by performing a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. Additionally, this nearby information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localised along an edge [44].

This step is carried out by expressing the difference of Gaussian image at the candidate keypoint location and scale, $D(x, y, \sigma)$, as a first order Taylor expansion:

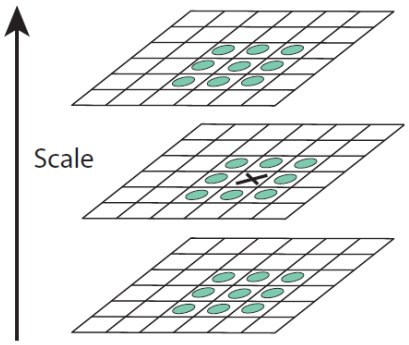


Figure 3.2: Identification of a SIFT feature as a maximum or minimum of the difference of Gaussian images by comparison with it its 26 neighbour pixels in 3x3 regions at the current and adjacent scales. Image obtained from [44].

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \tag{3.1}$$

where D and its derivatives are evaluated at the candidate keypoint location and $\mathbf{x} = (x, y, \sigma)^T$ is the offset from this point [44]. The sub-pixel location offset of the keypoint, $\hat{\mathbf{x}}$, is then obtained by taking the derivative of this function with respect to $\mathbf{x}$ and setting it to zero, which gives

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D}{\partial \mathbf{x}}. \tag{3.2}$$

3.1.3 Orientation Assignment

An orientation is assigned to each keypoint using local information from the Gaussian image, $L(x, y, \sigma)$, (not difference of Gaussian) at the same scale as the keypoint within a region centred on the feature point, according to the following equations:

$$m(x, y) = \sqrt{\{L(x+1, y) - L(x-1, y)\}^2 + \{L(x, y+1) - L(x, y-1)\}^2} \tag{3.3}$$

$$\theta(x, y) = \tan^{-1}\left(\{L(x, y+1) - L(x, y-1)\} / \{L(x+1, y) - L(x-1, y)\}\right) \tag{3.4}$$

where $m(x, y)$ is the gradient magnitude and $\theta(x, y)$ is the gradient orientation. These local gradient orientations are used to construct an orientation histogram of the sample points within the local region around the feature point. The histogram consists of 36 bins covering the 360 degree range of rotations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times the scale of the keypoint [44]. The peak value of the orientation histogram is assigned as the orientation of the feature point.

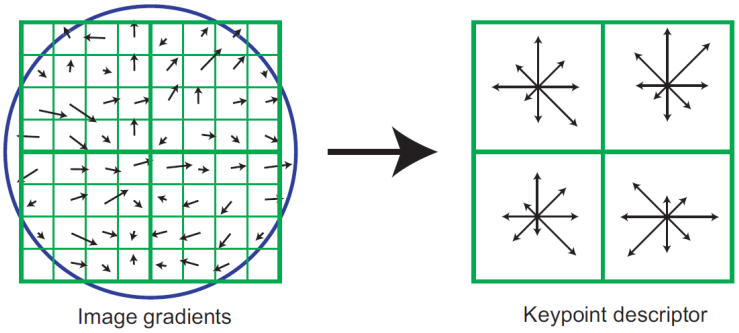


Figure 3.3: Computation of the SIFT Keypoint descriptor: An 8x8 image region is used to calculate image gradient magnitudes and directions for each pixel surrounding the keypoint, which are weighted using a circular Gaussian window (indicated by the circle). These are used to construct a 4x4 array of orientation histograms, which constitutes the feature descriptor for the SIFT keypoint. Image obtained from [44].

3.1.4 Keypoint Descriptor Calculation

So far the feature points have been assigned an image location, a scale and an orientation, which enable the definition of a local 2D coordinate system in which to describe the local image region surrounding the feature point and hence enable the keypoint to be invariant to scale and orientation. This step describes the calculation of a highly distinctive descriptor for the local image region that is robust to the remaining image variations of 3D viewpoint change, affine distortions, image noise, and changes in illumination in order to enable reliable matching across multiple images [44].

Using the scale of the feature point to determine which Gaussian image to use in the following, and the previously computed orientation to define a 2D coordinate system centred on the feature point, as mentioned in the previous paragraph, a 16x16 grid of image sample points is defined around the feature point. The image gradient and orientation of each of these sample points is computed using the equations presented in the previous subsection and these are weighted using a circular Gaussian window function to emphasise the sample points closer to the centre. The 16x16 grid of pixels is divided into 16 evenly sized sub-grids of 4x4 pixels. Each of these 4x4 pixel sub-grids is used to form an orientation histogram consisting of 8 bins over the full 360 degree range of orientations, where the magnitude of each bin is determined from the weighted sum of the magnitudes of all orientation vectors that fall within that bin. This results in a 4x4 descriptor array containing the 16, 8-binned orientation histograms, which is used to construct a 128 element feature descriptor vector for each SIFT keypoint. This method is summarised in Figure 3.3 for a simplified 2x2 descriptor array constructed from an 8x8 image region. The feature descriptor vector is then normalised to unit length to reduce the effects of illumination changes.

3.2 Application of SIFT to Descent Images

One of the advantages of the SIFT feature tracker algorithm reported in [44] was that it is capable of generating very large numbers of features that densely cover the image over the full range of scales and locations. This could be extremely beneficial for generating digital elevation models with the required accuracy and resolution for hazard detection since the requirements are for the ability to detect small surface hazards in order to enable a safe and precise pin-point landing on potentially hazardous terrain. Therefore a high density of tracked features will be required. This

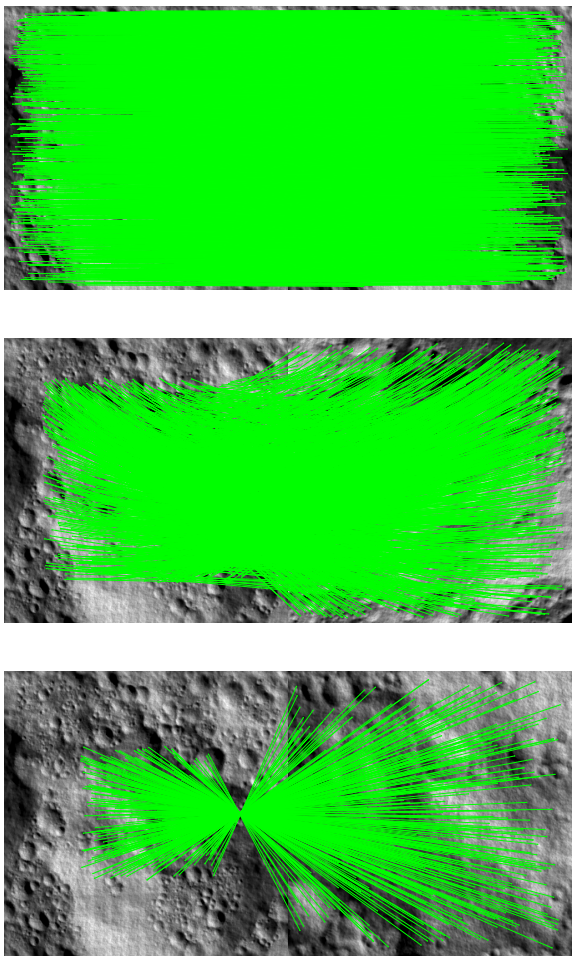


Figure 3.4: SIFT feature tracker applied to a 200 image PANGU descent sequence: Top – Feature matches between image 0 (left) and image 1 (right); Middle – Feature matches between image 0 (left) and image 100 (right); Bottom – Feature matches between image 0 (left) and image 199 (right)

claim of a high density of features is verified in Figure 3.4, where it can be seen that a very large number of feature points have been tracked reliably over a long image sequence.

The image dataset used in Figure 3.4 consists of 200 images with a frame rate of 20fps, created using the Planet and Asteroid Natural scene Generation Utility (PANGU), which is a software tool created for ESA by the University of Dundee for producing realistic terrain models and images of the types of terrain found on planetary, asteroidal, and natural satellite surfaces throughout the solar system. This image dataset represents a descent trajectory with pure vertical motion at a constant velocity of 100m/s and with a constant rotational velocity of 3rpm anticlockwise about the Z-axis. The ability of SIFT to extract and match a vast number of features is immediately apparent, especially for image 0 and image 1 (top of Figure 3.4), which contains 3459 feature matches. The features extracted from image 0 and successfully matched in image 1 are used to construct a feature database from which the features extracted in all

subsequent images are matched against. As time progresses and the rotation angle and scale change between image 0 and later images becomes more significant, fewer matches are able to be found. However it can be seen that a large number of matches are still found, with the middle image of Figure 3.4 showing 1896 matches between image 0 and image 100, and the bottom of Figure 3.4 showing 357 matches between image 0 and image 199, which is far superior to what can be achieved with other feature tracking methods, such as KLT (as will be seen below). This is a very encouraging result as it shows that a significant number of features should easily be able to be tracked for long enough periods of time to allow the weakly observable structure parameter estimates to converge to a suitable level of accuracy.

It would therefore appear that SIFT should be the feature detection and tracking method of choice for 3D reconstruction during planetary descent. However, it has been reported in [42] and [45] that SIFT features may not be the most suitable type of features required for reliable 3D reconstruction, as was demonstrated in a simple application in [45]. While a much more rigorous and extensive analysis of this possible unsuitability (than was given in [45]) would be required in order to determine the full implications of this potential problem, it must be noted that in this project it has not been possible to obtain any kind of reasonable results for the structure estimation using the SIFT feature tracker, which would appear to support the above mentioned claims. A full analysis of this problem is left to future work since it is beyond the scope of this project. In the remaining sections of this chapter, the KLT feature tracker is discussed, and used as an alternative to the SIFT algorithm despite it not being capable of providing the image feature density and longevity that was observed with SIFT. KLT features should, however, be more suitable for 3D reconstruction as they are predominantly corner type features, which are known to be reliable features for 3D reconstruction. It is also important to note that the KLT feature tracker is more computationally efficient than the SIFT feature detector, and so the use of KLT would be more conducive to a real-time system.

3.3 Conventional KLT

The conventional KLT algorithm was first proposed by Lucas and Kanade in 1981 [46] as an intuitive and straightforward means of tackling the image registration problem. The basic assumption is that if given an image I and a later image J with some unknown interframe motion, then a point (x, y) in image I has a corresponding point in image J that can be found by minimising some measure of the difference between $I(x, y)$ and $J(x + d_x, y + d_y)$ where (d_x, d_y) represents the image motion modelled as a straightforward displacement. This approach was further developed by Tomasi and Kanade in [47] and Shi and Tomasi in [29] to include a feature selection method that is optimal by construction because it is based on the way that the tracker works, i.e. the features are chosen to have the specific characteristics required for the tracker operate with maximum stability. Shi and Tomasi [29] also described a separate feature monitoring method that enabled errors in the tracking to be identified and thus allow for incorrectly tracked features to be removed from the tracking process so that their incorrectly determined locations would not corrupt the output of any higher level computer vision algorithms that depended on the tracking results. Due to the way that the tracking problem is formulated, it is required that the interframe motion is very small, which may not be a realistic assumption to make in many real world situations. To overcome this potentially restrictive requirement, Bouguet [48, 49] developed a pyramidal implementation of the KLT algorithm, in which image pyramids are constructed in order to artificially increase the scale of the images and therefore allow the small displacement requirement to be fulfilled at the largest scale in the pyramid. The tracking solution is iteratively refined over decreasing scales (increasing image size/resolution) by propagating the tracking solution down subsequent levels of the pyramid until the original full sized image is reached and the final tracking solution is obtained. This effectively allows for

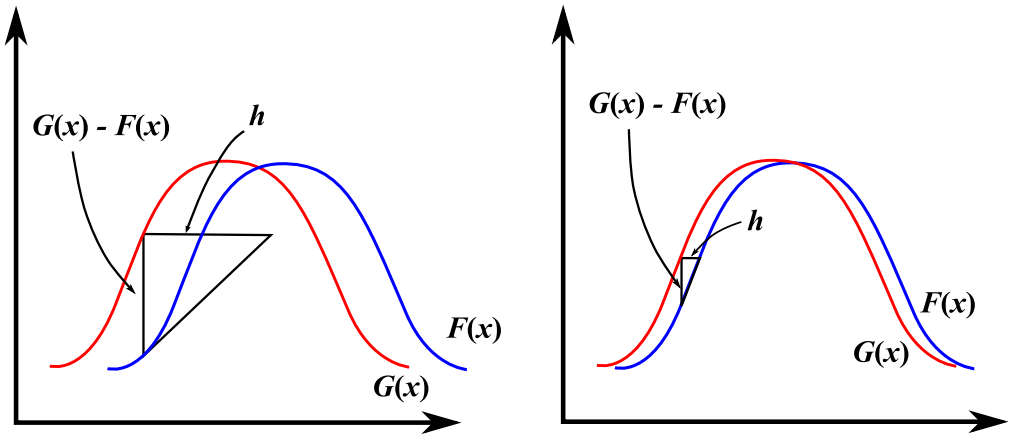


Figure 3.5: 1D simplification of the concept behind KLT

much larger displacements to be handled without problem. The pyramidal implementation of the KLT algorithm described by Bouguet is what we refer to as conventional KLT, and its formulation will be described below.

The basic premise of the KLT algorithm can be seen in Figure 3.5, which illustrates the 1D relationship between two identical curves, $F(x)$ and $G(x)$, separated by a some unknown displacement h. The left side of Figure 3.5 shows that when the displacement is too large and the tangent is taken at a point in the more non-linear part of the curve $F(x)$ is taken, the displacement h that completes the triangle formed by the difference between the two curves at the point x and the tangent line at the point x, does not represent the actual displacement between the two curves. However, when the displacement between the two curves is sufficiently small and the tangent is taken at a point within the more linear region of the curve, then the displacement h more accurately represents the actual displacement between the two curves, as shown on the right side of Figure 3.5. These two diagrams aptly describe the basic premise of the KLT method and the assumptions on which it is based – that the displacement is small and a linear approximation can be made. When these assumptions are valid, then in the 1D case, the displacement can be obtained by utilising the definition of the gradient, which is approximately given by

$$F'(x) \approx \frac{F(x+h) - F(x)}{h}, \tag{3.5}$$

recognising that $G(x) = F(x+h)$, the displacement is then given by

$$h \approx \frac{G(x) - F(x)}{F'(x)}. \tag{3.6}$$

Taking this simple 1D concept and extending it to 2D, allows the KLT tracking equation to be derived, which will be presented in the following subsection in the pyramidal form developed by Bouguet [48].

3.3.1 Pyramidal KLT Algorithm

Beginning with the assumption that the image motion can be represented by a small displacement, the following approximation can be made

$$I(x, y) \approx J(x + d_x, y + d_y). \tag{3.7}$$

Due to image noise and the possibility that the image motion is not completely described by a simple translation, it is important to consider this relationship over a 2D neighbourhood. Now, considering a point $\mathbf{u} = [u_x \quad u_y]^T$ in the first image I. The goal is to find the point $\mathbf{v} = \mathbf{u} + \mathbf{d} = [u_x + d_x \quad u_y + d_y]^T$ in the second image J. Therefore, the goal is to find the displacement vector $\mathbf{d}$ that minimises the residual function ϵ:

$$\epsilon(\mathbf{d}) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} \{I(x, y) - J(x + d_x, y + d_y)\}^2 \tag{3.8}$$

Now, if we define $I^0 = I$ to be the "zero$^{\text{th}}$" level of the image pyramid – i.e. the original, full sized image, where the image width and height for this level of the pyramid are defined as $n_x^0 = n_x$ and $n_y^0 = n_y$, respectively. The image pyramid is constructed in a recursive fashion, using the results of one level to compute next level – i.e. compute I^1 from I^0 and then I^2 from I^1, and so on up to the maximum pyramid level L_m. Thus, with $L = 1, 2, \ldots, L_m$

$$\begin{aligned} I^L(x, y) = \frac{1}{4} \; & I^{L-1}(2x, 2y) + \\ & \frac{1}{8} \left\{ I^{L-1}(2x-1, 2y) + I^{L-1}(2x+1, 2y) + I^{L-1}(2x, 2y-1) + I^{L-1}(2x, 2y+1) \right\} + \\ & \frac{1}{16} \left\{ I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x+1, 2y-1) + I^{L-1}(2x-1, 2y+1) + I^{L-1}(2x+1, 2y+1) \right\}. \end{aligned} \tag{3.9}$$

The image width and height for the pyramid level L are then given by the largest integer that satisfies the following expressions:

$$n_x^L \leq \frac{n_x^{L-1} + 1}{2} \tag{3.10}$$

$$n_y^L \leq \frac{n_y^{L-1} + 1}{2}. \tag{3.11}$$

With the image pyramid constructed the original point $\mathbf{u}$ is transformed to the point $\mathbf{u}^L$ in pyramid level L by

$$\mathbf{u}^L = \frac{\mathbf{u}}{2^L} \tag{3.12}$$

The overall pyramid tracking algorithm proceeds as follows: first the solution is computed at the top level of the pyramid L_m; this result is then propagated to the next level down $(L_m - 1)$ in the

form of an initial guess for the pixel displacement at that level; this initial guess is then iteratively refined for the level $L_m - 1$; this result is then propagated down to the next level ($L_m - 2$); and so on until the bottom of the pyramid is reached at which point the final solution is the full image displacement for the current image pair. Using $\mathbf{g}^L = [g_x^L \;\; g_y^L]^T$ as the initial guess that is available from processing the pyramid levels from level L_m to level $L + 1$, the residual equation that must be minimised is now given by:

$$\epsilon^L(\mathbf{d}^L) = \epsilon^L(d_x^L, d_y^L) = \sum_{x=u_x^L-w_x}^{u_x^L+w_x} \sum_{y=u_y^L-w_y}^{u_y^L+w_y} \left\{ I^L(x,y) - J^L(x + g_x^L + d_x^L, y + g_y^L + d_y^L) \right\}^2 . \quad (3.13)$$

Once $\mathbf{d}^L$ is computed by minimising this equation, the initial guess is propagated to the next level down through the use of the expression

$$\mathbf{g}^{L-1} = 2\left(\mathbf{g}^L + \mathbf{d}^L\right). \quad (3.14)$$

Note that no initial guess is available for the top level of the pyramid, but due to displacement being scaled down sufficiently such that the assumption of small displacement is valid, the initial guess can be given as

$$\mathbf{g}^{L_m} = [0 \;\; 0]^T. \quad (3.15)$$

At the bottom of the pyramid the final displacement is given by

$$\mathbf{d} = \mathbf{g}^0 + \mathbf{d}^0. \quad (3.16)$$

The solution the Equation (3.13) is obtained by differentiating with respect to d_x and d_y and setting equal to zero to obtain the minimum as well as approximating J by expressing it as a Taylor expansion truncated to the linear terms, to arrive at the equation

$$\mathbf{d}^L = (\boldsymbol{G}^L)^{-1}\mathbf{b}^L \quad (3.17)$$

which is solved iteratively for the displacement $\mathbf{d}^L$ in each pyramid level, where

$$\boldsymbol{G}^L = \sum_{x=u_x^L-w_x}^{u_x^L+w_x} \sum_{y=u_y^L-w_y}^{u_y^L+w_y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (3.18)$$

where I_x and I_y are the x and y image derivatives of image I^L, and

$$\mathbf{b}^L = \sum_{x=u_x^L-w_x}^{u_x^L+w_x} \sum_{y=u_y^L-w_y}^{u_y^L+w_y} \begin{bmatrix} \delta I I_x \\ \delta I I_y \end{bmatrix} \quad (3.19)$$

where $\delta I = I^L(x,y) - J^L(x + g_x^l + d_{x,k}^L, y + g_y^L + d_{y,k}^L)$, where k denotes the iteration step.

3.4 IMU-KLT

Conventional KLT has a potentially significantly limited operating range in that it is only reliable under small appearance changes from one image to the next. Thus it is only suitable in situations where the inter-frame camera motion is small. The operating range may be extended somewhat by employing a coarse-fine strategy via the use of image pyramids, as discussed above. However, such an approach needs to be implemented by taking into account the maximum expected pixel displacement, but even then it can only go so far and it is not adaptive in the event of sudden disturbances in the motion of the camera. Typically, pixel displacements greater than a few 10's of pixels would likely result in tracking failure.

The basic aim of the IMU-KLT algorithm [50, 51] is to build in a degree of robustness and adaptability to large image motions by using information supplied by an IMU. In the event of large, unexpected camera ego-motion the previous known feature location (used as the initial guess of the new feature location in standard pyramidal KLT) would likely be well outside of the solution convergence region, leading to a failure of the tracking algorithm. However, by using IMU measurements the location of the convergence region for each feature point can be predicted, thereby providing a much better initial guess for the new feature location, which will hopefully result in a successful convergence.

In real-time, video-rate tracking applications the main cause of large pixel displacements is large camera rotational motion [50]. Therefore, the IMU-KLT algorithm only makes use of gyroscopic measurements. However, it may be possible to further boost performance by also including translational acceleration measurements, but this is not investigated here, and so for the purpose of predicting the location of the convergence region for each feature point, it is assumed that the motion of each feature arises due to a pure camera rotation. In this case, the feature point motions can be described by a single 2D homography, $\boldsymbol{H}$:

$$\boldsymbol{H} = \boldsymbol{K}^{-1}\boldsymbol{R}\boldsymbol{K} \tag{3.20}$$

where $\boldsymbol{R}$ is the inter-frame rotation matrix computed using gyroscopic measurements, and $\boldsymbol{K}$ is a camera calibration matrix (for further details see [50, 51]). The rest of the IMU-KLT algorithm is similar to the standard pyramidal KLT technique as described in the previous section.

Figure 3.6 presents a comparison of the performance of IMU-KLT (right) with standard KLT (left) over a 100 image sequence using synthetic images generated from a 3D planetary surface model constructed using the Planet and Asteroid Natural Scene Generation Utility (PANGU). The top row shows the detected feature points in image 1 of the sequence, in which the initial number of features for the conventional KLT algorithm is 158 and for IMU-KLT it is 69. The middle row shows the number of features remaining at frame 35 in the sequence, where for standard KLT there are 30 features remaining and for IMU-KLT there are 68. The bottom row shows the remaining features at the end of the sequence. In this case there are only 14 features remaining for the standard KLT and 56 remaining for IMU-KLT. Given that conventional KLT was allowed to start with many more features and yet still ends up with far fewer features, it can be clearly seen that IMU-KLT offers far greater stability over conventional KLT. Thus, IMU-KLT would enable a significantly denser estimate of the scene structure. It is also interesting to note in this image sequence the rotational motion was relatively small (0.02rad/s on both the Y and Z axes), but even in this case the benefits of using inertial measurements are seemingly significant.

In implementing the IMU-KLT algorithm we have made some slight modifications to better suit our needs. The most significant of these modifications was the introduction of sub-pixel accuracy in the feature selection stage, the importance of which will be highlighted in Chapter 4. Another modification was the prevention of further feature extraction after the first image, since, to begin with (for initial testing purposes), we are only concerned with estimating the structure of the initial feature points and attempting to maximize the duration over which these initial features

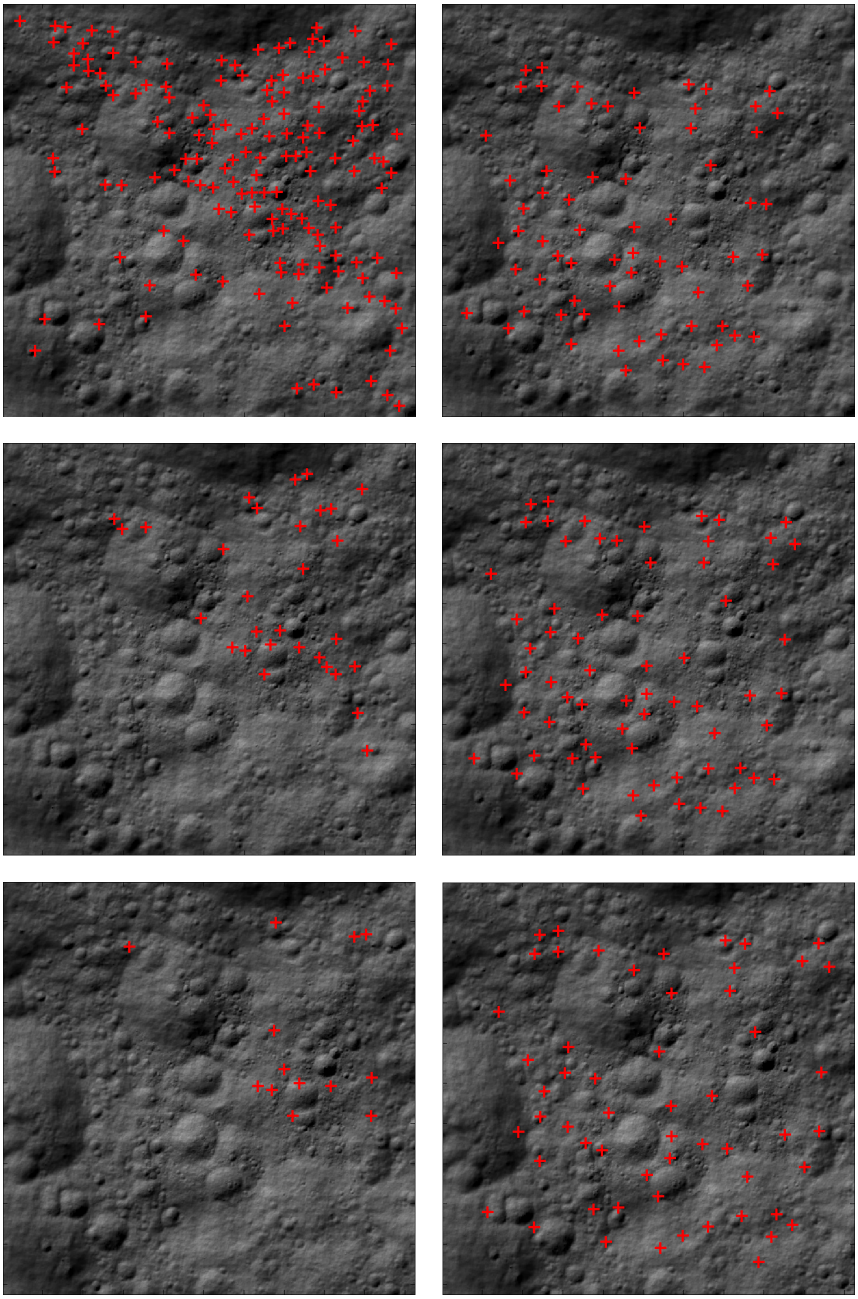


Figure 3.6: Performance comparison between IMU-KLT (right) and conventional KLT (left) over a 100 image sequence: Top – image 1; Middle – image 35; Bottom – image 99 (images are numbered 0–99)

can be tracked. We have also simplified the representation of IMU measurements to better suit the simulated IMU used in the early stages of the work carried out in this project.

The use of IMU-KLT in SFM was tested alongside a robust filtering method known as H_∞ filtering, and the results of this are presented in Chapter 5.

3.5 KLT with Time Reversibility Constraint

It has long been known that the conventional KLT algorithm suffers from an accumulation of errors that will eventually result in instability in the computation of the iterative tracking solution, which will ultimately lead to divergence and the loss of the feature. This can substantially limit the number of images over which a feature point can be reliably tracked. Even the IMU-KLT algorithm described above will still suffer from feature drift because, even though greater stability was observed, the introduction of IMU data is only used to provide an updated initial guess for the feature point position for the next image by shifting the tracking results from the previous image to a new location based on IMU measurements. However, the tracking results themselves will contain the accumulated errors from all previous steps. Additionally, because we were later advised (after preliminary results had been obtained using EKF and H_∞ filtering based SFM that also estimated motion to high accuracy) that the spacecraft's motion can be assumed to be known, we no longer had a need to model IMU measurements, and so IMU-KLT was not investigated further. Instead time was invested in a different formulation of KLT that is, at a fundamental level, more robust to the accumulation of errors.

This accumulation of errors manifests as a phenomenon known as feature drift, that even before the eventual loss of the feature point may introduce catastrophic errors in the calculations, such as SFM, that rely on the results of the feature tracking. The feature drift problem is illustrated in Figure 3.7 for a traditional corner-like feature, which shows two image sequences side by side. The left column illustrates feature tracking in which there is substantial drift that eventually causes the loss of the feature, whereas the right column shows feature tracking with no drift and this allows the feature to be reliably tracked for a longer duration. This issue with drift was acknowledged in [29] and was one of the motivations for developing the feature monitoring scheme that allowed erroneously tracked feature points to be discarded.

An innovative way of tackling the drift problem was developed by Wu et al [52] by realising that the problem is essentially down to a lack of any kind of constraint on the tracking solution. The approach can be elucidated by considering what might happen if the tracking was carried out in reverse time. If you took the results from a single forward-time tracking, i.e. the computed location of the feature point in image J at time t and used this as the starting point for tracking the feature point backwards in time to image I at time $t-1$, the result would likely not be the same as that which was obtained for the point in image I when the tracker was run in forward-time from $t-2$ to $t-1$. To overcome this problem a mathematical constraint was applied to the tracking equation to ensure that this time-reversibility property was fulfilled, without the need to actually run the tracker in reverse-time. The resulting algorithm can be called time-reversible KLT (TR-KLT).

The form of the residual equation, with this constraint imposed, is expressed as:

$$\begin{aligned}\epsilon(\mathbf{d},\mathbf{d^b}) = &\sum\sum_W \{J(\mathbf{u}+\mathbf{d}) - I(\mathbf{u})\}^2 + \\ &\sum\sum_W \left\{I(\mathbf{u}+\mathbf{d}+\mathbf{d^b}) - J(\mathbf{u}+\mathbf{d})\right\}^2 + \\ &\lambda\left(\mathbf{d}+\mathbf{d^b}\right)^T\left(\mathbf{d}+\mathbf{d^b}\right),\end{aligned} \tag{3.21}$$

where the first term is the same as the conventional KLT, the second term is the backwards tracking term, and the third term is the forwards, $\mathbf{d}$, and backwards, $\mathbf{d^b}$, displacement constraint. Differentiating this equation with respect to $\mathbf{d}$ and $\mathbf{d^b}$, setting equal to zero to obtain the minimum, and using first order Taylor expansions for $I(\mathbf{u}+\mathbf{d}+\mathbf{d^b})$ and $J(\mathbf{u}+\mathbf{d})$, a solution can be obtained in the form

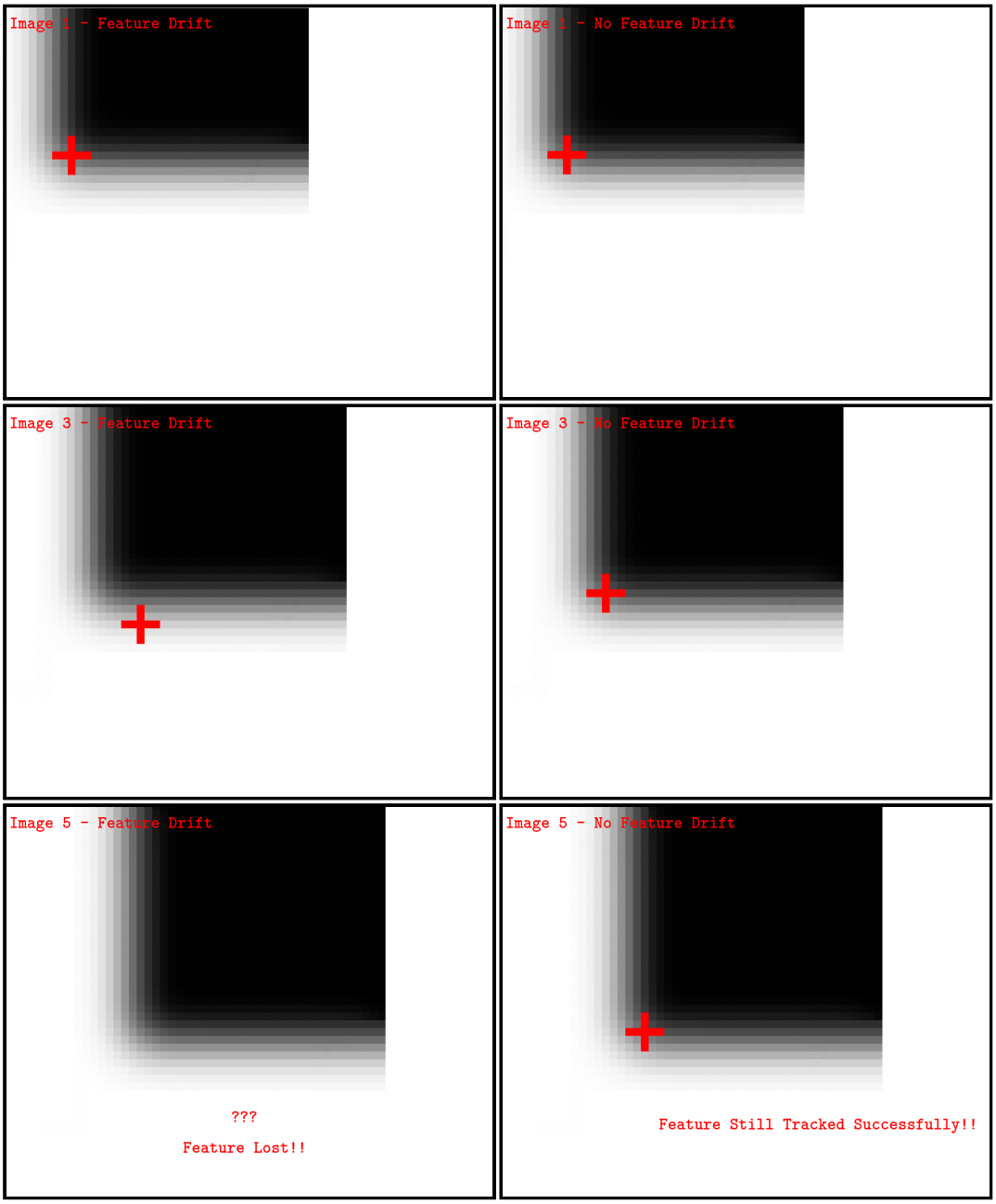


Figure 3.7: Drift in conventional KLT: Left column – Feature drift resulting in feature loss; Right column – Tracking with no drift allowing longer tracking duration

$$\mathbf{d} = \boldsymbol{U}^{-1}\mathbf{e} \tag{3.22}$$

where

$$\boldsymbol{U} = \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{C} + \lambda\boldsymbol{D}^{-1}\boldsymbol{C} - \frac{1}{2}\boldsymbol{B}, \tag{3.23}$$

$$\mathbf{e} = (\boldsymbol{A} + \lambda\boldsymbol{I})\,\boldsymbol{D}^{-1}\,(\boldsymbol{V} - \boldsymbol{W}) + \frac{1}{2}\,(\boldsymbol{S} - \boldsymbol{R})\,, \tag{3.24}$$

and where

$$\begin{aligned}
\boldsymbol{A} &= \sum\sum_{W}(\nabla I)^T\nabla I & \boldsymbol{B} &= \sum\sum_{W}(\nabla I)^T\nabla J \\
\boldsymbol{C} &= \sum\sum_{W}(\nabla J)^T\nabla J & \boldsymbol{D} &= \sum\sum_{W}(\nabla J)^T\nabla I \\
\boldsymbol{R} &= \sum\sum_{W} I(\nabla I)^T & \boldsymbol{S} &= \sum\sum_{W} J(\nabla I)^T \\
\boldsymbol{V} &= \sum\sum_{W} I(\nabla J)^T & \boldsymbol{W} &= \sum\sum_{W} J(\nabla J)^T
\end{aligned} \tag{3.25}$$

3.6 Performance Comparison

In order to assess the performance of this more recent and sophisticated approach to KLT feature tracking, which is described more fully in [52], it will first be compared against the conventional KLT in a simple test of tracking stability, using a 100 image sequence of synthetic descent images, to see how many features remain at the end of the sequence. Once this straightforward comparison has been made, a more thorough analysis of the two KLT methods will be made using multiple sets of synthetic images with various different types of image features present and with varying levels of image noise in order to assess the accuracy of the two methods by measuring the tracking errors with time compared to ground truth feature locations.

Figure 3.8 presents a side-by-side comparison of the two techniques for the 100 image synthetic visible-band sequence, with the conventional KLT tracking results in the left column and the TR-KLT tracking results in the right column. The top row shows the number of features that are extracted from image 1, from which it can be seen that the number of initial features are similar. The bottom row shows the remaining features in image 100. From this it can clearly be seen that the TR-KLT provides significantly better performance in terms of tracking stability through observing the number of features that are remaining – 65 for TR-KLT compared to 13 for conventional KLT. This is a very dramatic difference and in fact it was surprising to observe just how stark the difference in performance was.

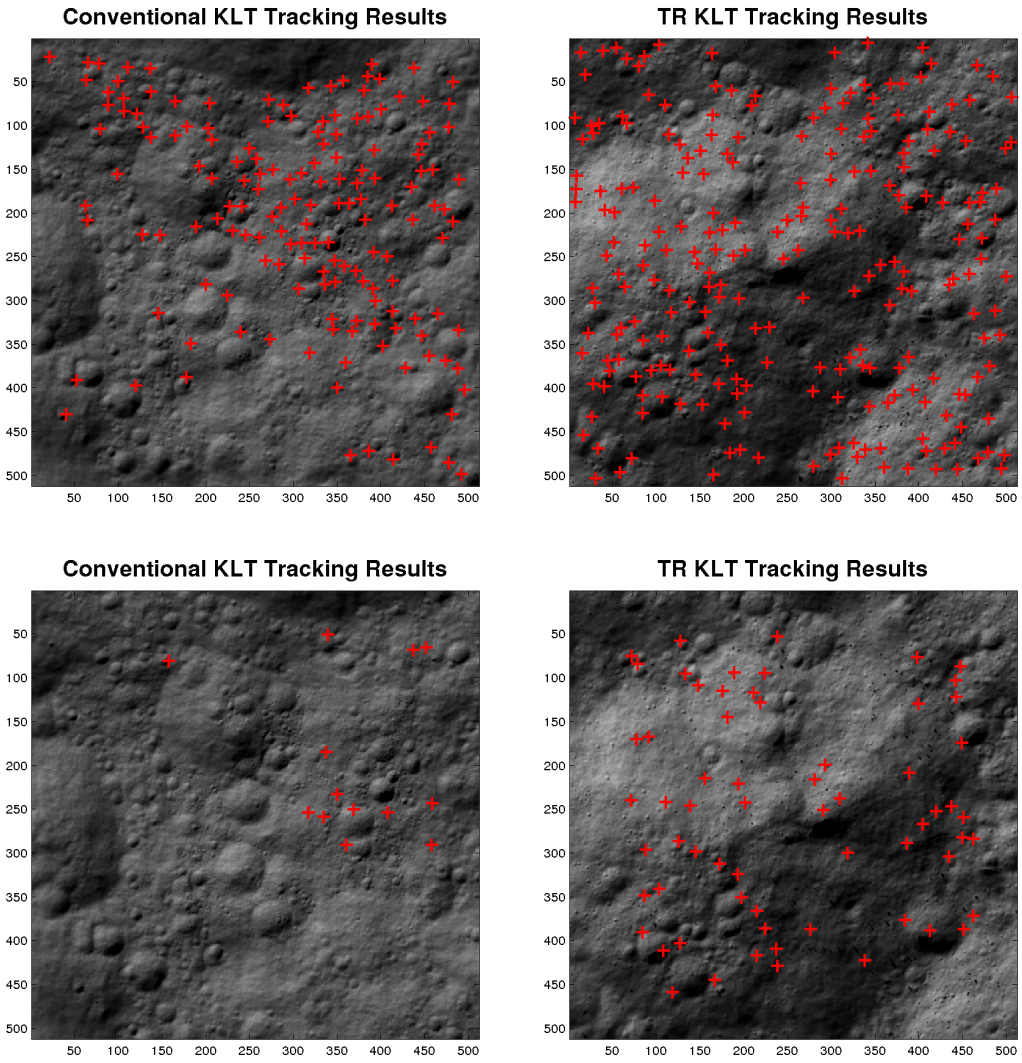


Figure 3.8: Comparison of TR-KLT with conventional KLT: Top Row – Features selected from image 1; Bottom Row – Features remaining in image 100

Figure 3.9 shows the pattern used in analysing the tracking accuracy of the two different KLT feature tracking algorithms. This pattern consists of a pure black and white chequerboard with 32x32 pixel squares, a number of Y shaped objects consisting of a central black line of single pixel width encased within a one pixel width grey outline, and a number of Gaussian points with a diameter of 7 pixels. This pattern was chosen because it would produce a collection of well defined and easily detectable feature points for which ground truth feature coordinates are easily obtainable. To construct the image datasets, this pattern is inserted into the top left (64 pixels away from the edge to allow for the pyramidal scaling) of a 1024x1024 pixel image with a pure white background, a slight Gaussian image blur operation is applied, and then for each subsequent image the pattern is shifted 5 pixels to the right and 5 pixels down and random Gaussian image noise is added with zero mean and differing variance values for each dataset. Each dataset consists of 106 images, which, with the amount of image displacement used, results in a final

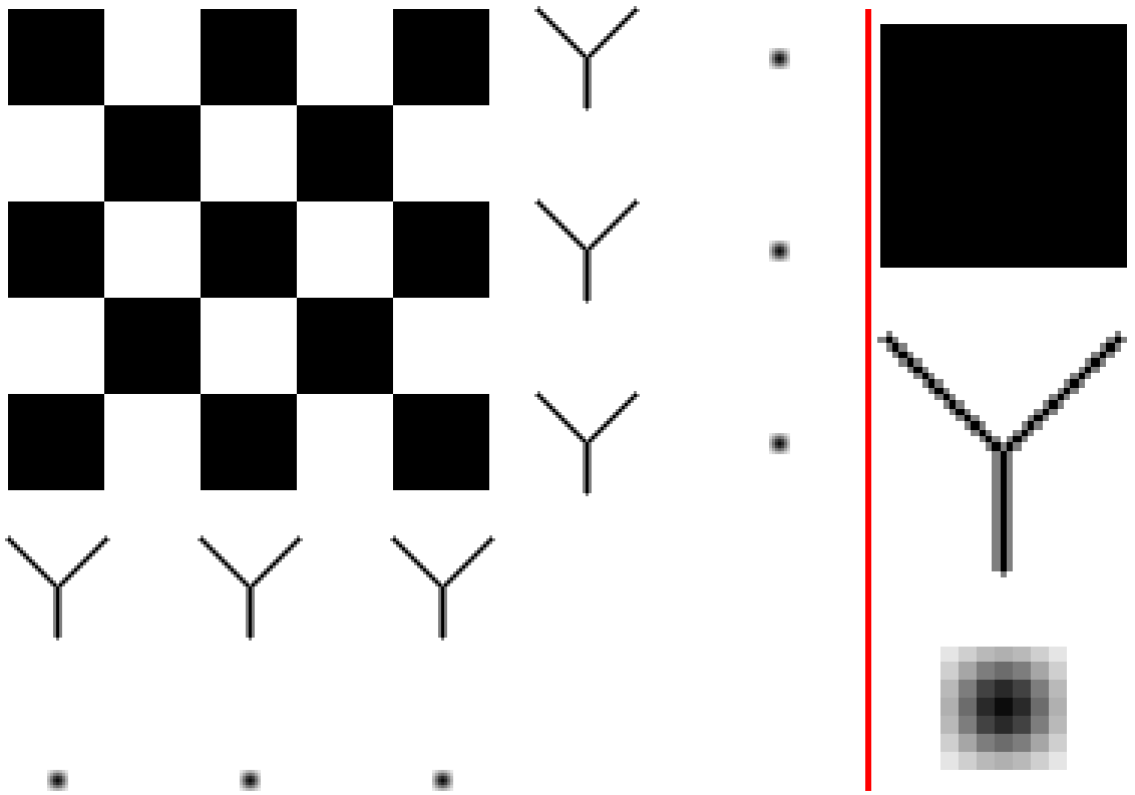


Figure 3.9: Synthetic feature pattern used in testing the accuracy of TR-KLT and conventional KLT: Left of red line – Feature pattern; Right of red line – Zoomed views of the different feature types: chequerboard square (top), Y shape (middle), Gaussian point (bottom)

image containing the pattern in the bottom right corner with a suitably sized margin. The first image in each of the sequences used in this analysis consists of an unblurred image with no noise to ensure that the initial feature selection for both KLT algorithms results in identical feature points that are uncorrupted by noise. Figure 3.10 shows a selection of the images used in this analysis, with various levels of image noise.

To quantify the performance of the conventional KLT and TR-KLT algorithms, the Euclidean distance of each point from its corresponding ground truth location is calculated to give the tracking error for each point. The average tracking error for each image is then obtained by computing the root-mean-squared tracking discrepancy over all feature points in the image. A plot of these RMS errors has been produced for each version of KLT for a single trial of each of the datasets to enable a direct comparison of the two techniques. The results of this analysis are presented in Figures 3.11, 3.12, and 3.13.

Figure 3.11 presents the RMS tracking errors over the 106 image sequence of both the conventional KLT and TR-KLT algorithms, where the images (except for the first image) are corrupted by zero-mean white noise with an intensity variance of 0.0001. A trend line is fitted to each set of results using linear regression. This plot shows that on the whole the performance of the two algorithms are very similar in terms of tracking errors, with the trend being that the errors do not appear to be significantly growing with time for either tracking algorithm. It can be seen that the blue plot for the TR-KLT shows slightly smaller errors, which is more clearly shown by observing the trend line, suggesting that TR-KLT gives slightly better tracking accuracy, however

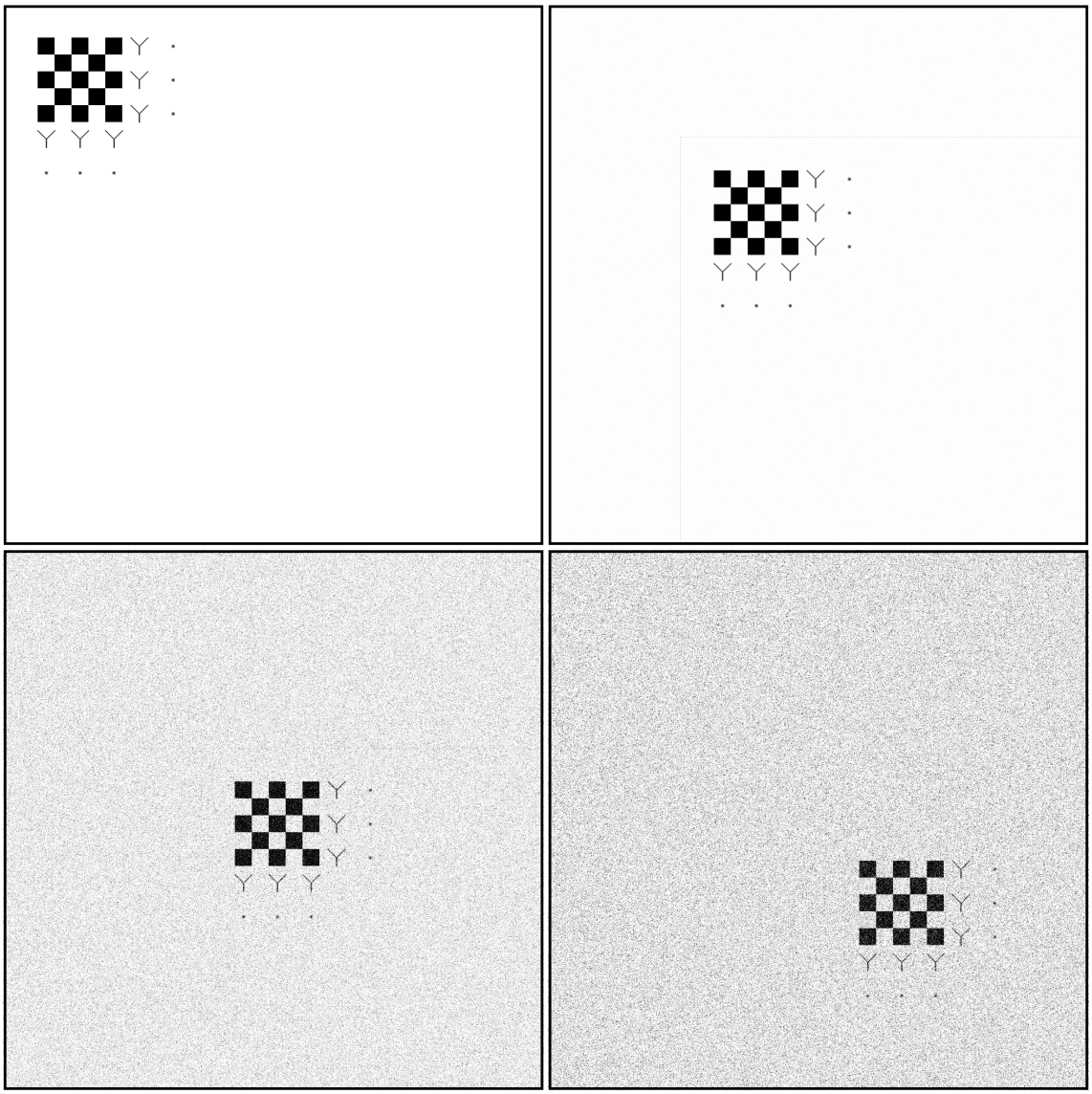


Figure 3.10: Example images used in testing KLT accuracy: Top Left – Image 0 (no noise); Top Right – Image 50 (noise variance = 0.0001); Bottom Left – Image 75 (noise variance = 0.05); Bottom Right – Image 105 (noise variance = 0.1)

the difference is not particularly significant in this case. Further tests were performed using the 106 image sequence with noise variances of 0.0005, 0.001, and 0.005, but these resulted in very similar results to the 0.0001 variance case, and so these results are not included here. It was not until noise variances of 0.01 and above were used that significant differences in performance were observed. The results for the 0.01 noise variance dataset are presented in Figure 3.12, which shows a clear difference in tracking accuracy from the two algorithms. Both tracking methods begin with the same initial error, but the TR-KLT quickly shows stronger performance compared to the conventional KLT. Trend lines have also been fitted to these plots using linear regression, which clearly show that the conventional KLT is suffering from growing errors in tracking accuracy, which confirms that conventional KLT can fall victim to the feature drift

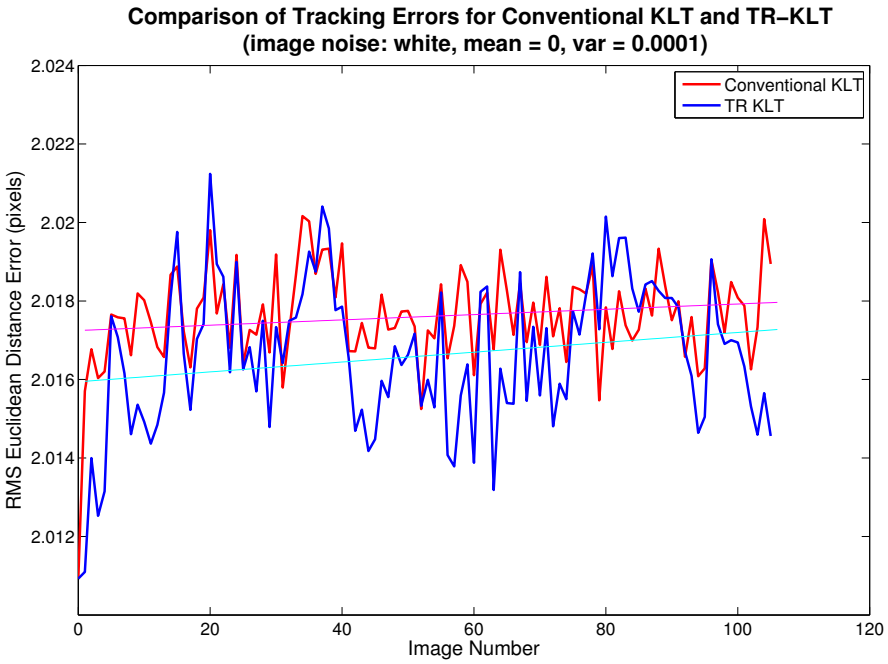


Figure 3.11: RMS tracking errors for conventional KLT and TR-KLT (image noise variance = 0.0001)

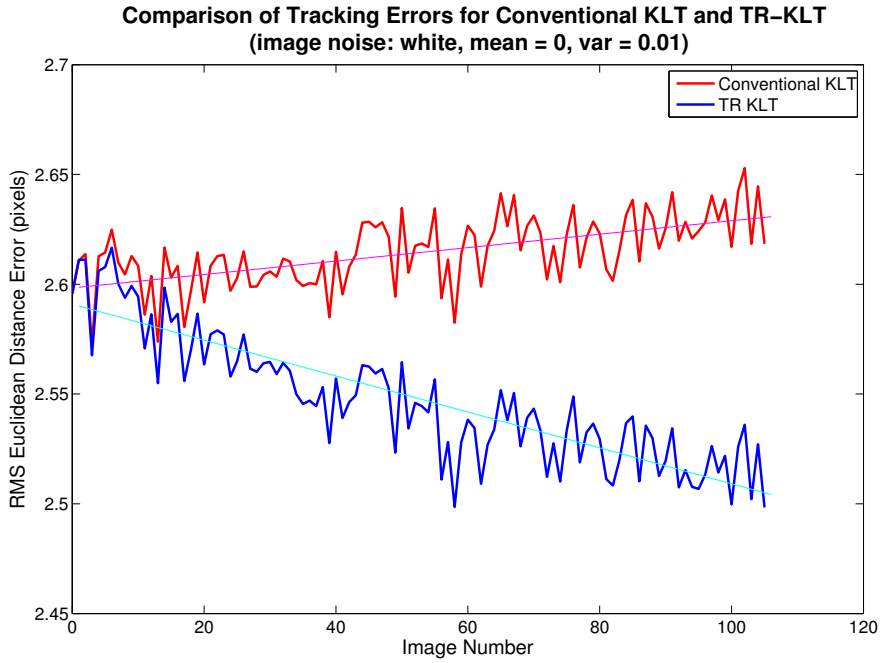


Figure 3.12: RMS tracking errors for conventional KLT and TR-KLT (image noise variance = 0.01)

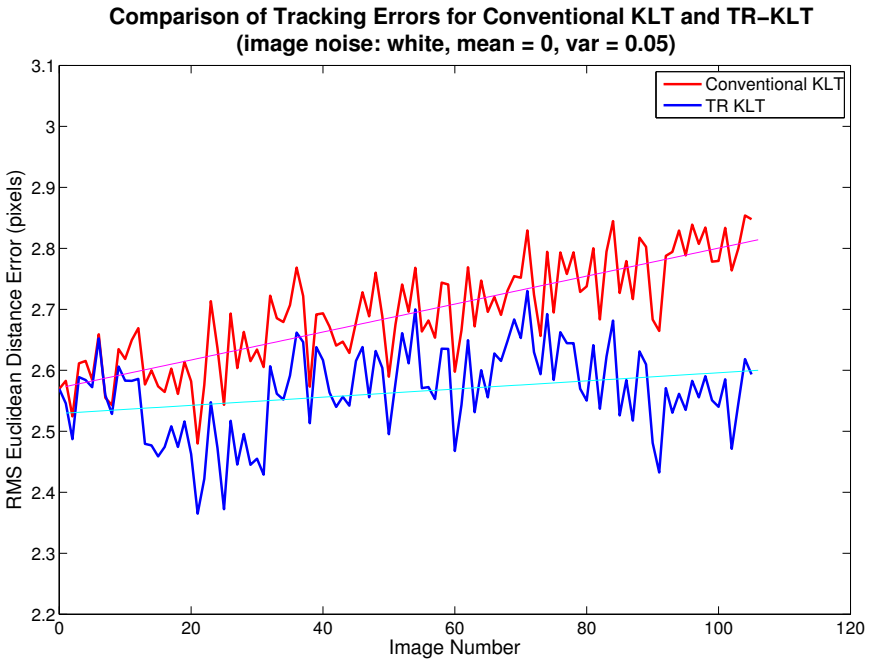


Figure 3.13: RMS tracking errors for conventional KLT and TR-KLT (image noise variance = 0.05)

problem discussed earlier. The trend line for TR-KLT shows that the errors are actually decreasing as time progresses. It would be expected that the tracking errors would either remain approximately constant or exhibit a slight growth with time (due to the constraint only extending back in time by 1 image) that is less pronounced than with the conventional KLT, therefore this would seem to be a somewhat unexpected result. However this can be explained by the fact that it was observed that the feature points for the Y features and Gaussian points were gradually lost over the image sequence, eventually leaving only the chequerboard features, which result in much stronger and hence more stable features, resulting in a reduction in average feature tracking errors over the sequence. A similar gradual loss of these features was observed for the conventional KLT tracker, however in this case the errors can be seen to still gradually increase with time, which clearly demonstrates the superior performance of TR-KLT over conventional KLT in this case. Figure 3.13 presents the rms tracking errors for the image sequence with noise variance of 0.05. In this case, for both algorithms, the weaker features on the Y shaped objects and the Gaussian points were lost almost immediately, and so in this case the results are almost solely due to the stronger chequerboard features. Consequently, the results are more in line with expectation in that they show growing tracking errors for both algorithms. However, for the TR-KLT algorithm this is much less severe than with the conventional KLT, which is exhibiting a comparatively very strong growth with time, indicating the susceptibility of this unconstrained algorithm to much more significant feature drift. A further test was carried out using the image sequence with noise variance of 0.1, one image of which is presented in Figure 3.10, however it was found that the level of noise in these images was too severe for reliable feature tracking over any extended length of time for both feature tracking methods, therefore a meaningful comparison could not be made in this case.

From the analysis presented above, the advantages of the TR-KLT in terms of feature drift and consequently feature stability over time are quite clear to see in comparison to the conventional approach to KLT feature tracking. This feature method will therefore be used in

favour of conventional KLT (in Chapter 6 onwards) as a suitable method for increasing robustness and accuracy in the feature measurements taken from the descent images and which are, in turn, used in the feature based structure from motion algorithm.

3.7 Conclusions

This chapter has presented the concepts and implementation details of two popular methods for extracting and matching/tracking image features over sequences of images: the SIFT feature matching algorithm and the KLT feature tracker. The SIFT algorithm was shown to be capable of tracking vastly more image features than the KLT algorithm, however it was realised, through the inability of the SIFT algorithm to successfully produce accurate structure estimates when used in SFM, that SIFT features are often not suitable for monocular 3D reconstruction. In this thesis, 3 different variants of the KLT algorithm are utilised in SFM, and the details of these variants were outlined in this chapter. These 3 variations of KLT are the conventional pyramidal KLT, IMU-KLT, and TR-KLT. The IMU-KLT algorithm was able to successfully track many more features than conventional KLT, but as will be seen in Chapter 5, it was not sufficiently accurate to produce reliable structure estimates. The conventional KLT algorithm is successfully utilised in Chapter 4, but as was shown in this chapter it is only capable of successfully tracking a small number of features for any significant length of time. It was concluded that this may likely be due to the accumulation of errors in the tracking solution over time, for each feature point, that not only would introduce errors in structure estimation, but eventually leads to loss of the feature point due to divergence. To combat this, an alternative formulation of the KLT tracker was investigated that introduces constraints in the tracking equations to ensure time-reversibility of the solutions. This chapter successfully demonstrated that this modified version of the KLT algorithm is able to track a larger number of features for longer periods of time and results in much lower error accumulation in the feature points over time in comparison to the conventional KLT algorithm.

4 STRUCTURE FROM MOTION

This chapter presents a multi-source, multi rate, recursive data fusion algorithm that aims at combining a single camera, recursive, feature-based structure from motion (SFM) algorithm with measurements from an on-board inertial measurement unit (IMU), using the Extended Kalman filter (EKF). The chapter describes how the fusion of these measurements enables direct estimation of the unknown scale factor that arises from the use of a single camera, and results are presented showing that this leads to an accurate estimation of the motion parameters and structure of the scene. The algorithm is tested using simulated IMU measurements and artificially generated image feature points with simulated image motion throughout a 100 image sequence (referred to as the 'pure synthetic' test) as a means to validate the formulation of the algorithm. Sequences of artificial images are also used that are generated from an artificial terrain model of a planetary surface, using the Planet and Asteroid Natural Scene Generation Utility (PANGU), to provide a more realistic test on representative images. These tests using PANGU images are carried out from three different initial heights above the surface (2000m, 1000m and 500m), which represent challenging scenarios for the verification of the algorithm. The SFM algorithm presented in this chapter is seemingly unique amongst the reported vision based EDL methods in the literature for both motion and structure estimation in that it is capable of using a single camera to produce fully-scaled metric estimates of both motion and structure without the need of any prior information (e.g. known landmarks). However, this chapter makes a number of simplifying assumptions, since at this stage the focus is on algorithm development rather than true realism in the test scenarios. These assumptions are: that the spacecraft descends at a constant velocity, with this velocity being representative of that of the NASA Curiosity rover's descent profile (for the initial altitude of each small test image sequence only); that the attitude of the camera during capture of the first descent image of each sequence is such that the camera frame is perfectly horizontal (Z-axis points in the nadir direction); and that the motion of the spacecraft is entirely in the vertical direction. In addition to these some slight rotational motion is modelled with constant angular velocity in order to partially simulate swinging motion under a parachute and some residual spin from spin-stabilisation. Further to this, only short image sequences are used so that there is no need to add new image features over time, in order to simplify the assessment of the structure estimation by allowing all features to have the same amount of time to approach convergence. Section 4.1 of this chapter gives a review of the different approaches to structure from motion in the literature. Section 4.2 gives details and assumptions relating to the type of landing scenario adopted in order to provide a test for the developed algorithm. Section 4.3 gives an overview of the general system and defines the coordinates systems used in the work. Section 4.4 describes the formulation of the recursive feature-based structure from motion algorithm. Section 4.5 describes how the scale is recovered through the fusion of SFM and IMU measurements, with Section 4.6 discussing the sequence in which these two measurement sources are combined. Section 4.7 then describes how the results of the estimation process can be used to obtain the absolute structure parameters and construct a DEM of the landing site that could (with further development) be used in Hazard detection. Section 4.8 presents the results from the pure synthetic test and the results of three different test trajectories using PANGU images. Finally, the conclusions are presented in Section 4.9.

4.1 Review of Structure from Motion Techniques

The way in which 3D objects are projected onto a 2D image plane over a sequence of images in which there is relative motion between the scene and camera not only depends upon the relative 3D motion but also on the 3D scene structure. Thus the image sequence contains information that can enable both the structure and motion to be recovered. The structure and motion are generally inseparable, i.e. one cannot be recovered without knowledge of the other. Therefore, these two properties are often estimated simultaneously.

The class of techniques that focus on the recovery of both the scene structure and the relative motion are known as structure from motion (SFM) algorithms. The SFM problem has been an active area of research for around 40 years, and over this time there have generally been two main approaches to addressing the problem: feature-based approaches that utilize a set of sparse image features; and optical flow-based approaches that attempt to determine the 2D image motion at every pixel and from this estimate the 3D motion and structure of the scene [53].

Optical flow methods begin by examining the instantaneous changes in image brightness values from one image to the next in order to estimate a dense velocity map known as the image flow or optical flow field [54]. This flow field essentially consists of a velocity vector for each pixel in the image pair. To achieve this, these techniques rely on the local spatial and temporal derivatives of image brightness values under the assumption that the brightness value recorded in the images associated with a particular point in the scene remains constant between the images. Thus any change in brightness of an image pixel (x, y) from one image to the next is due only to the relative motion between the scene and camera. That is, the whole brightness pattern recorded by the camera is assumed to have been translated due to the relative motion.

Once the optical flow field has been determined, the 3D structure and relative motion between the scene and camera can be estimated using the information contained in the optical flow. This is achieved by relating the optical flow field to the camera model and the motion of the camera, from which a system of equations can be derived in terms of the optical flow, its first- and second-order derivatives, and the 3D structure and motion parameters [54]. The system of equations consists of 12 equations in 11 unknowns and is therefore over determined. However, the equations are non-linear and so a unique solution is not possible. A large number of different techniques have been reported in the literature for solving these equations, which often involve restricting the nature of the motion to be purely translational or rotational and/or restricting the imaged surface to be planar (see refs [53–55] for a summary of some of the available methods).

Given that the optical flow field is based on derivatives, optical flow measurements are inherently noise sensitive. Additionally, since there is potentially a measurement for every pixel in the image, algorithms that utilize optical flow can be very computationally demanding unless the flow-field is sub-sampled [55]. Thus optical flow techniques may be difficult to implement in real-time applications. The majority of the techniques for flow based SFM are also formulated as two-frame methods [56, 57] where a pair of images are analysed to give an instantaneous estimation of the scene structure and the between frame motion. Extending this over multiple frames to determine a full motion trajectory involves integrating the results from successive pairs of images, which may lead to significant error accumulation due to the noise in the derivative parameters [58]. However, more modern techniques have been developed to address the issues with noise, such as using flow probability instead of estimating the optical flow directly as in [59], or using an extended Kalman filter (EKF) in a recursive framework that utilizes affine flow parameters within local regions, that are estimated using linear regression, as measurements instead of direct optical flow [55].

Feature-based approaches to the SFM problem are based on extracting a set of relatively sparse but highly discriminatory, two-dimensional features in the images corresponding to three-dimensional object features in the scene, such as corners, occluding boundaries/depth discontinuities, and boundaries demarcating changes in surface reflectivity [54]. These features

are extracted from each image and inter-frame correspondence between the features is established. Once a set of corresponded features have been identified, the 2D image locations of these features can be used along with the camera model equation and motion models to estimate the 3D structure and motion parameters. For reviews of feature based techniques see [53, 54, 60, 61].

The early work in feature based SFM was mainly concentrated on two or three frame approaches. The first of this type of technique was that of Ullman in 1979 [62], who showed that under the assumption of rigidity and orthographic projection, a set of equations could be written relating the unknown 3D distances between the points to their known 2D image projections [63]. Each image would result in a different set of equations relating the same object points, and so given enough images and enough points the equations could be solved for the unknown 3D structure. Ullmann demonstrated that the structure could be recovered with at least three views of four points and that the solution was closed form [63]. A more general approach by Roach and Aggarwal [64] used the rigidity assumption to formulate the problem as a modified least squares error method to solve the system of non-linear equations resulting from the use of perspective projection. They found that two views of six points or three views of four points were needed to provide an over determined set of equations. However, it was discovered that the method was not very accurate unless considerably more points were used in the images. Another approach to the two-frame problem is the so-called 8 point algorithm of Longuet-Higgins [65], and the similar algorithm of Tsai and Huang [66]. The Longuet-Higgins 8-point algorithm [65] determines the rotation and translation between the two views from the essential matrix, which is derived using vector and tensor analysis, and then uses these motion parameters to determine the 3D structure (depths) of the points. A similar method was independently developed by Tsai and Huang [66], and was shown that in many cases the motion parameters, and thus the structure, can be uniquely determined, given eight image point correspondences.

The two-frame approaches to feature-based SFM are derived for perfect features and for images with wide baselines. As such they do not consider measurement errors in the feature point correspondences. The presence of noise in the images often leads to violations of some of the key assumptions on which these techniques are based, therefore they can exhibit numerical instability in real-world situations [61]. The results obtained also only give a snapshot of the motion and structure parameters. In many applications motion estimation results are required over extended periods of time in order to provide an estimate of the full trajectory of the camera or imaged objects. To address this issue, a great deal of work has been reported in the literature to extend these techniques to long sequences of images, such as image streams from a video camera. For more information see [53, 54, 60, 61].

There are two main approaches to implementing this extension to multiple images: batch techniques; and recursive techniques. Batch techniques often use a non-linear framework similar to the classic relative orientation problem proposed by Horn [67], and perform a batch minimization, such as a Levenberg-Marquardt non-linear minimisation, over the whole set of measurements [61]. Thus, as each new image arrives, it and all previous images are processed to compute the motion and structure estimates. Clearly, therefore, such techniques can be very computationally demanding for large image sequences. In practice, this is a technique that would not be applied as each new image arrives but would be applied intermittently on relatively large sets of images; therefore it is not suitable for real time operation. Recursive techniques, on the other hand, process each new image individually as it arrives, and sequentially update a previous estimate of the motion and structure parameters. One way of achieving this is to repeat the classic two-frame estimate, such as the Longuet-Higgins technique [65], and integrate the results over time. Examples of this type of approach are the work of Oliensis and Thomas [68] and Soatto et al [69]. However, given that image sequences are typically recorded at relatively high frame rates, the baseline between successive images will be small; therefore the results from each application of a classic two-frame approach will be prone to significant error. This

issue was addressed in [68] and [69] by the use of a smoothing Kalman filter. Other recursive approaches make use of the extended Kalman filter (EKF) to directly estimate the structure and motion expressed in state-space form, instead of using the Kalman filter simply for smoothing two-frame estimates [61]. These methods do not make use of two-frame techniques.

The first recursive EKF-based SFM algorithm was that of Broida and Chellappa [70], which focused on the estimation of motion parameters of a rigid body undergoing unknown rotational and translational motion, using measurements of noisy image coordinates of two or more object correspondence points. The advantage of using the EKF framework is that the unknown parameters are not a function of the number of image frames used, unlike the two-frame type methods where the number of unknowns increases with the number of images used. Therefore over-determination of the estimation equations can be achieved by using a large number of frames, instead of a large number of feature points [70]. Another advantage is that since the evolution of the state is governed by the kinematics model, the algorithm is still able to propagate the state information in the event that the object becomes partially or even fully occluded. A number of significant assumptions were made to simplify the estimation task: the structure was assumed in order to fix the scale factor to enable absolute translational position and motion to be estimated; the motion was constrained so that the dynamic equations were fully linear; and the image match points and their measurement noise were assumed to be known a priori, thus no attempt at feature tracking was made. This work was later extended to a more physically realistic scenario in a later paper by Broida, Chandrashekhar and Chellappa [71], where in this case the constraints on the motion were relaxed leading to non-linear motion equations, and now both the structure and motion were estimated. However, the feature match points were still assumed to be available. This algorithm also made use of a batch technique carried out on the first few frames in order to initialise the EKF, which was demonstrated to lead to improved performance. Azarbayejani and Pentland [72] improved upon the work of Broida, Chandrashekhar and Chellappa [71] by reformulating the problem in terms of a slightly different geometrical representation of the camera model, which allows the structure to be represented by a single parameter in the state equation. This representation has the important property that the camera is decoupled from the depth, which means that the model does not become numerically ill-conditioned as focal length becomes large, allowing both the orthographic and perspective cases, and anything in between, to be modelled without changing the representation. Additionally, this also allows the focal length to estimated as well as the motion and structure parameters, which means that knowledge of the camera need not be known in advance. This formulation was shown to significantly increase the robustness and accuracy of the structure and motion estimates.

While the early work on recursive feature based techniques focused on establishing the theoretical underpinnings and general formulation of the problem, more recent work has focused on addressing the issues encountered when applying the techniques in real-world applications. One problem that has been looked at regularly in the literature is that of erroneous feature point correspondences, known as outliers, which occur when using real-imagery as a consequence of image noise and imperfect feature matching techniques. The presence of such outliers in the measurement data can have devastating consequences for the results obtained from recursive estimation algorithms. Examples of work in this area can be found in [73–76]. Another area that has been investigated is the incorporation of inertial data to improve robustness in SFM algorithms. Due to the properties of the EKF and the way that it is implemented in traditional model-based SFM algorithms, robust results can generally only be obtained when the motion varies smoothly. Non-uniform camera egomotion introduces system modelling error or makes the algorithms converge to values other than the correct motion parameters. The addition of inertial information enables more complicated motion to be handled and has been shown to decrease the inherent biases and variances in motion parameter estimates [53]. It has also been shown that a smaller number of features are required to obtain accurate and robust results when inertial

data is exploited [77]. Further examples of SFM algorithms that employ inertial data can be found in [78, 79]. While the work in these examples exploited inertial information as a means for improving the robustness and accuracy of the overall SFM results [77, 78], or exploited SFM as a means of correcting for drift in the inertial measurements [79], Nutzi et al [80] showed that the use of inertial measurements from a 3-axis accelerometer could actually be used to recover the unknown scale factor in a single camera simultaneous localization and mapping algorithm, which is a similar technique to SFM. The scale factor issue is implicit in all of the above mentioned SFM algorithms, and arises due to the fact that any scalar multiple of the three scene point coordinates results in the same image point. It is therefore impossible to determine the depth of points in the scene using a single camera without further information, and therefore impossible to determine absolute motion and structure. Recovery of the scale factor therefore enables recovery of absolute translational motion, position and structure. The direct recovery of the scale factor was also shown to offer improvements in robustness and accuracy in general. Finally, improvements have also been sought through the use of different types of recursive filtering methods. Venter and Herbst [81] proposed a recursive solution based on the unscented Kalman filter, which was found to outperform the EKF methods when no initial data was available, however in all other situations tested it did not offer any improvements over the conventional approaches. Clipp et al [75] implemented a recursive solution using a pipelined pair of EKFs acting on the same data but offset in time. The leading filter produced estimates of structure and motion from all the available data, which were used to identify the best set of feature measurements to be used in the following filter in order to improve the accuracy of the results. Although encouraging results were obtained, this arrangement introduces a delay in the system, which may not be suitable for certain applications, such as when the estimation results are used in a closed-loop control system in which response time is critical, thus is unlikely to be acceptable in the system proposed in this work. The vast majority of approaches to recursive feature-based SFM algorithms have been based on the extended Kalman filter, or at least variants of it. The reasons for this are most likely because the EKF is widely known and a well studied estimation technique. However, the EKF is known to have limitations that are attributed to the underlying assumptions of the Kalman filter, such as the assumption of zero-mean, uncorrelated Gaussian noise, which may not be valid in all situations. Additionally, the EKF linearises about the current state estimate and covariance in each time step using a first order Taylor expansion. However, when the deviation of the estimated state trajectory from the nominal state trajectory is large, the abandoned higher order terms will contain large values, giving rise to considerable linearisation error that significantly degrades the performance of the estimator. These limitations prompted research, carried out in this department, into reformulating the recursive feature-based structure from motion problem using an estimator based on the L_∞-norm, instead of the L_2-norm, which does not suffer from the same limitations [82]. Experiments were carried out on real and synthetic images, comparing the algorithm with a traditional EKF approach based on the formulation of Azarbayejani and Pentland [72], and significant improvements were observed.

The two main approaches to structure from motion (optical flow/feature-based) are vastly different in nature and utility in the way in which they treat the concept of structure. Chellapa et al [53] suggest that optical flow-based techniques are conceivably better suited to modelling the 3D scene, while the feature-based approaches are more suitable for estimating the 3D motion parameters of the camera rather than the scene structure. Feature based techniques are also better suited to applications in which it is expected that individual features remain visible for extended periods of time and can thus be tracked over multiple images [83]. However, in our work we require both highly accurate estimates of the motion parameters as well as an accurate and dense 3D digital elevation model of the terrain surrounding the landing site. Research has been reported in the literature that attempts to combine the two techniques in order to achieve both of these characteristics – see [58, 84] for example. However, in this current chapter we will be focusing mainly on the accurate estimation of the 3D motion parameters as this is arguably

the most significant step in the development of a next-generation EDL since it is a prerequisite for all of the other required capabilities. Additionally, since we assume that the landing site will be visible for the full duration of the decent and for the majority of the decent image features will not move rapidly out of the visual field, we adopt a feature based structure from motion technique using a filtering framework. Therefore we are focusing on motion and sparse structure estimation, leaving the development of a hybrid approach to further work. The motion parameters must be estimated in real-time as the ultimate goal is for their use in a closed-loop control system to guide the lander to its target landing site, therefore we require a recursive algorithm. We also require the motion and structure parameters to be known in absolute quantities, and given that inertial measurement units are common place on first generation EDL, we chose to adapt the data fusion strategy of Nutzi et al [80] to the SFM problem in order to directly recover the scale factor, and also due to its claims on increased robustness. We have adopted the recursive algorithm of Azarbayejani and Pentland [72] for our SFM framework in order to provide a straight forward verification of our approach, but with the view of extending this work to the more robust formulation presented in [82] at a later date.

4.2 Landing Scenario

For the purpose of the work presented in this chapter, a number of assumptions are made relating to the landing scenario, while at the same time ensuring that the situation remains somewhat representative of that which may be encountered. These assumptions are as follows: it is assumed that the spacecraft will emerge from the entry phase at a point that is above the intended landing site such that the landing site will be in the field of view of the camera from the beginning; the motion of the camera/spacecraft is assumed to be relatively smooth and stable such that the landing site remains in the field of view for the duration of the descent; the velocity at the start of the motion estimation process is known in advance (the reason for this assumption will be discussed below); given that the current trend is towards larger, more capable missions, the choice of velocity at a particular altitude is loosely based on the expected velocity profile of the Mars Science Laboratory mission, presented in [85]; during the 100 image sequence it is assumed that the velocity of the spacecraft remains constant, and any deviation will be modelled as noise; and, the camera system and motion estimation process will be activated at a height of 2000m, and will operate continuously until touchdown.

4.3 Coordinate Systems and General System Overview

Figure 4.1 shows the coordinate systems used in this work. The position and orientation of the camera frame at the time that the first image is captured is used to define the world frame. The world frame is shown at the top of Figure 4.1. All motion and structure is described relative to this world frame, that is, the translation and rotation of the camera frame in any subsequent image (bottom coordinate frame in Figure 4.1) is defined with respect to the initial position at the start of the motion estimation process. The structure parameters are the 3D coordinates of the image features identified in the first image, expressed in the camera frame coordinates of this first image. With the motion estimation beginning at 2000m, the first image will cover a large area surrounding the landing site, and as the spacecraft descends, the structure of the scene captured in this image will be recursively refined leading to a large 3D terrain model. Using this large model, a potentially long list of alternative landing sites could be selected if the originally intended one is deemed too hazardous. A large DEM with many possible landing sites identified introduces a high degree of flexibility to the system that should contribute significantly to the reliability of the system.

Image features are identified in the first image and tracked across the rest of the image

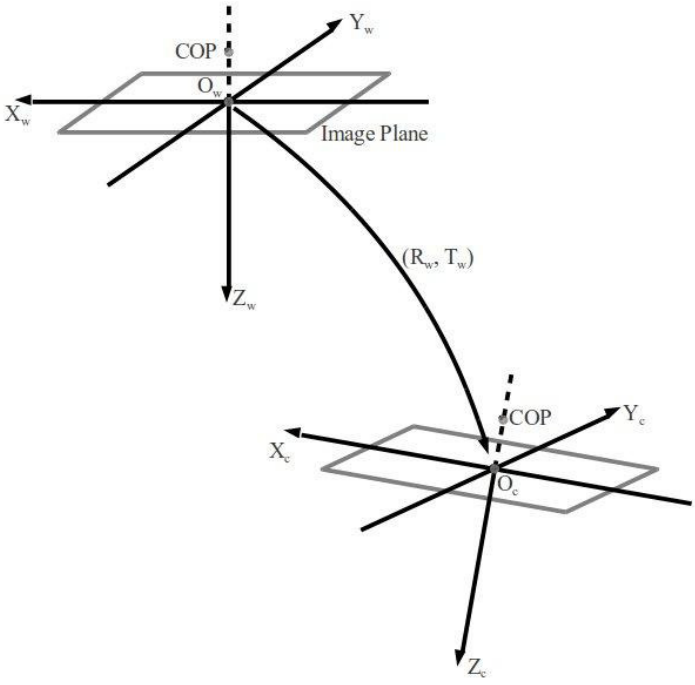


Figure 4.1: Coordinate systems: world frame (top), later camera frame (bottom). Global rotation matrix $\boldsymbol{R}_w$ describes the rotation about the $\mathbf{X}_w$, $\mathbf{Y}_w$, $\mathbf{Z_w}$ coordinate axes required to rotate the world frame into the orientation of the current camera frame. The incremental rotation matrix (see later) also represents incremental rotations about the world frame axes. The translation vector, $\mathbf{T}_w$, describes the displacement of the current camera frame with respect to the world frame, expressed in the world frame coordinate system

sequence using the conventional Kanade-Lucas-Tomasi (KLT) feature tracking method. The image plane coordinates of these initial features are used in conjunction with the current estimate of the spacecraft's motion to calculate the measurement predictions in each iteration of the SFM algorithm. Therefore, it is important that the original features are detected to sub-pixel accuracy. For sub-pixel detection the method described in [42] is used to fit a paraboloid surface to the Harris interest strengths of the pixels in a template window surrounding each interest point. To achieve the necessary accuracy for motion estimation and structure recovery the features must also be tracked to sub-pixel accuracy. For sub-pixel tracking the bi-linear interpolation method described in [48] is adopted.

The original KLT method requires that the inter-frame motion of the camera is small. This

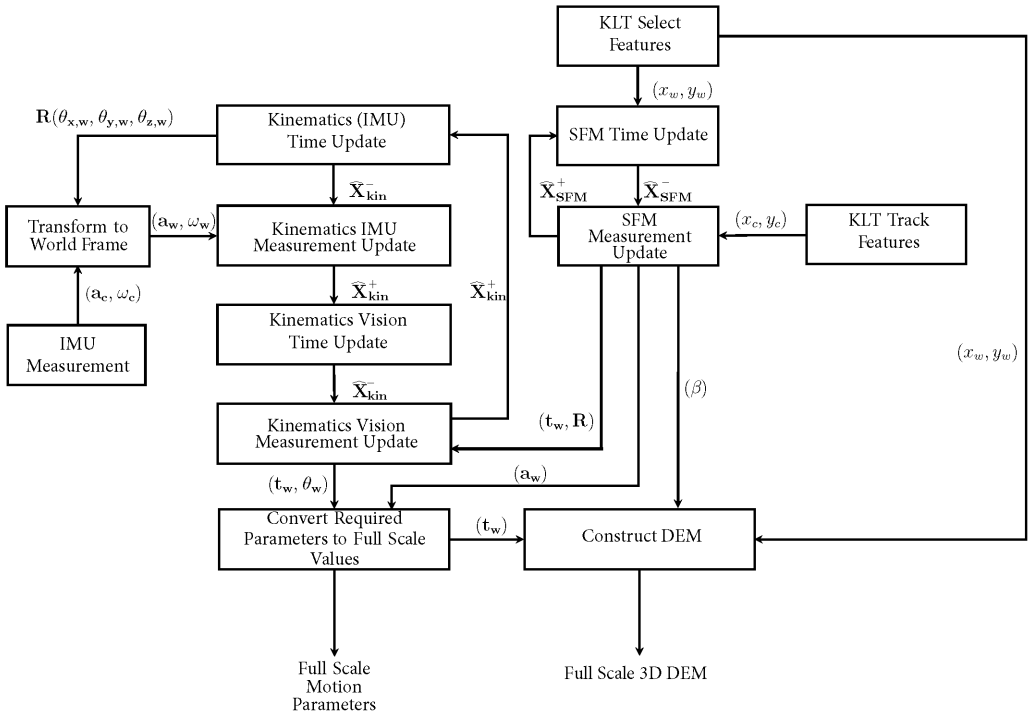


Figure 4.2: System diagram

can be quite restrictive for real world applications, particularly in the case of a planetary landing which may be subject to shocks and sudden movements due to high winds, control thrusters, and potential swinging under a parachute. To overcome this the pyramidal implementation of KLT developed by Bouguet [48] is employed, which utilizes image pyramids to increase robustness by artificially reducing the inter-frame motion. The KLT algorithm is a well known feature tracking method, thus it will not be described here. For details the reader is referred to [29, 46–48] and Chapter 3.

The SFM algorithm provides estimates of the 3D motion parameters of the spacecraft as it descends towards the surface. However, these parameters can only be determined up to an unknown scale since only a single camera is employed. The reason for this scale ambiguity is due to the loss of depth information as a consequence of the image formation process, as was explained in Chapter 2. However, by fusing these estimates with measurements from an IMU, which contains information about the absolute translational motion of the spacecraft, the unknown scale factor can be directly estimated, and therefore the absolute motion parameters and absolute depth in the scene can be recovered. To do this, two separate EKFs are used, operating at different rates. One EKF is used to recursively estimate 3D motion parameters using measurements from an IMU operating at 160Hz. In this EKF the translational motion is expressed as an unscaled quantity by dividing through by the unknown scale factor, thus allowing it to be directly related to the translational motion in the SFM algorithm. A separate EKF is then used to estimate the unscaled structure and motion parameters from images captured at 40 frames per second (the SFM algorithm). These parameters are then used as measurements in the first EKF in a separate measurement update operation in order to fuse the data from the two separate sources. Note that since a separate filter is employed to perform the data fusion of IMU measurements with the SFM filter estimates, ringing may occur when one filter converges

before the other. It is therefore necessary to take care when selecting suitable tuning parameters for each of the separate filters to ensure that the effects of this are minimised as best as possible. A potential way of determining whether this phenomena is occurring would be to examine whether the estimation of the scale factor is well behaved as a function of time and shows strong signs of convergence to a sensible value. Figure 4.2 presents a schematic diagram summarizing the operation of the overall system. Full details of the two separate modules and the data fusion process are given in the following sections.

4.4 Structure From Motion Algorithm

The structure from motion algorithm developed in this work is categorized under the framework for recursive recovery of structure and motion as well as focal length. The camera model used in this algorithm is shown in Equation (4.1), where (X_c, Y_c, Z_c) is the 3D location of a point in the camera frame, (x, y) is the image location, and $\beta = 1/f$ is the inverse focal length. This camera model equation is easily derived through similar triangles by considering the geometry shown in Figure 4.3. Note that the origin of the camera frame is at the principal point rather than the centre of projection (COP), as would normally be the case in the regular perspective camera model. The authors of [72] point out that this form of camera model has the property that the representation of the camera (i.e. β) is decoupled from the depth (Z_c), which is important when focal length is being estimated in addition to structure. Also this model does not become numerically ill-conditioned as the focal length becomes large (i.e. it is also suitable for orthographic projection models, where $f = \infty$, in which case $\beta = 0$).

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X_c \\ Y_c \end{bmatrix} \frac{1}{1 + \beta Z_c} \tag{4.1}$$

By noting that the Z-component of the image plane is zero, Equation (4.1) can be rearranged to give an expression relating the full 3D camera frame coordinates of a point in the scene to the coordinates of the image of that point on the image plane:

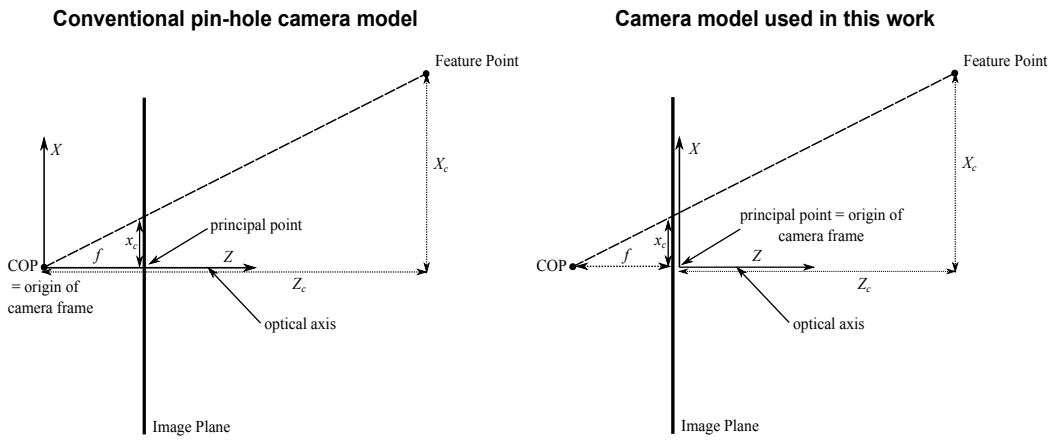


Figure 4.3: Camera model geometry

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \\ 0 \end{bmatrix} + Z_c \begin{bmatrix} x_c\beta \\ y_c\beta \\ 1 \end{bmatrix} \tag{4.2}$$

However, as mentioned above, the scene structure that is required is the 3D location of feature points identified in the first image, i.e. in the world coordinate system. Thus Equation (4.2) can be re-written as

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} x_w \\ y_w \\ 0 \end{bmatrix} + \alpha_w \begin{bmatrix} x_w\beta \\ y_w\beta \\ 1 \end{bmatrix} \tag{4.3}$$

where in this case $Z_w = \alpha_w$. The first term, in this equation represents the image location and the second term represents the perspective ray scaled by the unknown depth α_w. Therefore point-wise structure can be represented with a single parameter per point [72]. Thus a set of N feature points identified in the first image results in the set of structure parameters $\{\alpha_{1,w},\ \alpha_{2,w},\ \cdots,\ \alpha_{N.w}\}$.

In this work it is desirable to describe the translational motion as the 3D location of the camera frame with respect to the stationary world frame (in world frame coordinates), which could be represented by the vector

$$\mathbf{t}_w = \begin{bmatrix} t_{X,w} & t_{Y,w} & t_{Z,w} \end{bmatrix}^{\mathrm{T}} \tag{4.4}$$

However, in [72], the translational motion is defined such that it describes the 3D location of the world frame with respect to the current camera frame (in current camera frame coordinates). This situation arises due to the measurements being carried out in the current camera frame, making it more straightforward to directly estimate the motion quantities expressed in camera frame coordinates. Nevertheless, it is still straightforward to describe the motion in terms of the parameters that are of most use in this current application. Thus an equation can be written that relates the 2D coordinates of the feature points as measured in the current camera frame to the original 2D features in the world frame and the translation of the current camera frame with respect to the world frame:

$$\begin{bmatrix} x_c(1+Z_c\beta) \\ y_c(1+Z_c\beta) \\ Z_c\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \beta \end{bmatrix} \delta\boldsymbol{R} \cdot \boldsymbol{R} \begin{bmatrix} x_w(1+\alpha_w\beta) - t_{X_w} \\ y_w(1+\alpha_w\beta) - t_{Y_w} \\ \alpha_w - (t_{Z_w}\beta)/\beta \end{bmatrix}, \tag{4.5}$$

where the subscript c denotes the current camera frame and subscript w denotes the world frame; $\boldsymbol{R}$ represents a global rotation matrix describing a rotation of the coordinate axes from the world frame to the current camera frame. Note that $\boldsymbol{R}$ is treated as invariate within each estimation step and an incremental rotation matrix $\delta\boldsymbol{R}$, derived from incremental Euler angles, is approximated using the following small angle approximation:

$$\delta\boldsymbol{R} = \begin{bmatrix} 1 & \delta\psi & -\delta\theta \\ -\delta\psi & 1 & \delta\phi \\ \delta\theta & -\delta\phi & 1 \end{bmatrix}. \tag{4.6}$$

The global rotation matrix, $\boldsymbol{R}$, is updated following each measurement update step to incorporate the estimated incremental rotation, to give the new global rotation to be used in the next time-step.

The translation vector $\begin{bmatrix} t_{X_w} & t_{Y_w} & (t_{Z_w}\beta) \end{bmatrix}^T$ describes the translation of the current camera frame with respect to the world frame in world frame coordinates. The minus signs in Equation (4.5) arise because a translation of the camera by $\mathbf{t_w}$ results in the feature points appearing to move by $-\mathbf{t_w}$ in the scene. Note also that t_{Z_w} has been scaled by the inverse focal length in order to overcome the fact that there is much less sensitivity to t_{Z_w} motion [72]. The factor $t_{Z_w}\beta$ appears in parenthesis to indicate that this would be estimated (if estimating motion as well as structure) as a single parameter and not as a product of two separately estimated parameters.

By representing the global rotation matrix in the form

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{4.7}$$

Equation (4.5) can be used to derive an expression for the 2D location of the i^{th} feature point on the image plane for any image in the sequence, $\begin{bmatrix} x_{i,c} & y_{i,c} \end{bmatrix}^T$, in terms of the image coordinates of the features in the world frame and the known motion parameters at the time the image was captured:

$$\begin{bmatrix} x_{i,c} \\ y_{i,c} \end{bmatrix} = \frac{1}{1+Z_c\beta} \begin{bmatrix} A\left(x_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{X_w}\right)+B\left(y_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{Y_w}\right)+C\left(\alpha_{i,w}-\left(t_{Z_w}\beta\right)/\beta\right) \\ D\left(x_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{X_w}\right)+E\left(y_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{Y_w}\right)+F\left(\alpha_{i,w}-\left(t_{Z_w}\beta\right)/\beta\right) \end{bmatrix} \tag{4.8}$$

where

$$Z_c\beta = G\left(x_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{X_w}\right)+H\left(y_{i,w}\left(1+\alpha_{i,w}\beta\right)-t_{Y_w}\right)+I\left(\alpha_{i,w}-\left(t_{Z_w}\beta\right)/\beta\right)$$

and

$$\begin{aligned} A &= \left(r_{11}+r_{21}\delta\psi-r_{31}\delta\theta\right), & B &= \left(r_{12}+r_{22}\delta\psi-r_{32}\delta\theta\right), & C &= \left(r_{13}+r_{23}\delta\psi-r_{33}\delta\theta\right), \\ D &= \left(r_{21}+r_{31}\delta\phi-r_{11}\delta\psi\right), & E &= \left(r_{22}+r_{32}\delta\phi-r_{12}\delta\psi\right), & F &= \left(r_{23}+r_{33}\delta\phi-r_{13}\delta\psi\right), \\ G &= \left(r_{31}+r_{11}\delta\theta-r_{21}\delta\phi\right)\beta, & H &= \left(r_{32}+r_{12}\delta\theta-r_{22}\delta\phi\right)\beta, & I &= \left(r_{33}+r_{13}\delta\theta-r_{23}\delta\phi\right)\beta. \end{aligned}$$

Therefore, Equation (4.8) is the measurement prediction equation that can be used in a filtering framework to predict the image coordinates of the feature points in each image of the sequence.

Since we aim to estimate both the structure and motion parameters, the state vector used in this estimation framework consists of $7+N$ parameters – six for the motion of the camera relative to the world frame, N structure parameters corresponding to the N feature points extracted from the first image (world frame), and the inverse focal length, β, of the camera. Therefore the state vector can be written

$$\mathbf{X}_{SFM} = \begin{bmatrix} t_{SFM,X,w} & t_{SFM,Y,w} & \left(t_{SFM,Z,w}\beta\right) & \delta\phi_{SFM} & \delta\theta_{SFM} & \delta\psi_{SFM} & \beta & \alpha_{SFM,1\ldots N,w} \end{bmatrix}^T \tag{4.9}$$

where the subscript SFM is used to denote that this is the state vector for the structure from motion EKF.

The motion of a planetary lander as it descends through an atmosphere can potentially be quite complex since at any point there may be variable thrust retro-rockets firing, complex pendulous motion under a parachute, wind buffeting, etc. Thus, a specific dynamics model is not specified for use in the time-update step of the structure from motion EKF. Only, the addition of noise to account for any motion that may occur is considered in the time update, in the form of the process noise covariance matrix Q.

4.5 Scale Recovery

Structure from motion algorithms using a single camera suffer from the problem that translational motion and structure can only be recovered up to an unknown scale factor. This problem arises due to the fact that a scene point can be at any arbitrary depth along the same perspective ray and still give the same (x, y) coordinate on the image plane, or put another way, any scalar multiple of the 3 scene point coordinates results in the same image. It is therefore impossible to determine the depth of points in the scene without further information, as was discussed in Chapter 2.

To overcome this problem, and enable structure from motion algorithms to run, it is common practice to set a single parameter to an arbitrary value in order to fix the scale. In two frame problems this is often achieved by fixing the length of the baseline between the two cameras i.e. fixing the magnitude of the translational motion between each image frame. It is argued in [72] that this is not a good practice because if the motion between two frames is ever zero at some point, then the estimation becomes numerically ill-conditioned. Additionally, since motion is generally dynamic, fixing the magnitude of inter-frame translation to some constant value for all frames results in the scale changing from frame to frame. Instead, it is suggested to fix a static parameter such as one of the structure parameters by setting its initial variance to zero in the EKF, which results in all the other parameters automatically scaling themselves with respect to this value [72]. While this approach was demonstrated to increase robustness, it is still not a good practice to adopt since this requires that particular feature to be visible across the entire sequence [58], which is unlikely to happen in practice. In this work the need to arbitrarily fix the scale is avoided altogether by directly estimating the true scale factor.

Inspired by the work of Nutzi et al [80], vision and IMU data are fused in an EKF in order to directly estimate the unknown scale factor. An IMU provides measurements of acceleration that are given as fully scaled quantities. These acceleration measurements can be integrated to give information on the absolute trajectory of the spacecraft as it descends towards its landing site. Fusing this information with the un-scaled motion estimates from single camera SFM allows for the direct estimation of the scale factor, which may then be used to give absolute estimates of the motion and structure parameters. Acceleration data is acquired from a 3-axis accelerometer, as was also done in [80]; however this work also makes use of measurements from a 3-axis rate gyro to improve the accuracy of rotational motion estimation. As indicated above, IMUs

are common place in current first generation EDL systems Thus, it is reasonable to assume that the use of IMU data will have almost no impact on the EDL system design for future missions since these instruments are included almost by default.

To incorporate the information from the IMU to address the scale ambiguity, a simple kinematics model of the spacecraft's motion is adopted, using the following time update equations:

$$\begin{aligned} \mathbf{t}_k &= \mathbf{t}_{k-1} + \mathbf{v}_{k-1}T + \frac{1}{2}\mathbf{a}_{k-1}T^2 \\ \mathbf{v}_k &= \mathbf{v}_{k-1} + \mathbf{a}_{k-1}T \\ \mathbf{\Theta}_k &= \mathbf{\Theta}_{k-1} + \boldsymbol{\omega}_{k-1}T \end{aligned} \tag{4.10}$$

where $\mathbf{t}$ is the translation vector, $\mathbf{v}$ is the translational velocity vector, $\mathbf{a}$ is the translational acceleration vector, $\mathbf{\Theta}$ is the total rotation Euler angle vector, $\boldsymbol{\omega}$ is the rotational velocity vector, T is the time step duration, and subscript k denotes the current time step number.

To make a distinction between the EKF used for structure from motion and the EKF used to incorporate IMU measurements, the subscript KIN will be used to indicate kinematics. Equations (4.10) show the underlying kinematics model that we use for the kinematics EKF. However there is a subtle difference between the quantities represented in these equations and the quantities in the state vector. Namely, incremental velocity is used instead of total velocity as this was found to give greater stability. The state vector for the kinematics EKF is as follows:

$$\mathbf{X}_{KIN} = \begin{bmatrix} \mathbf{t}_{KIN,w} & \delta\mathbf{v}_{KIN,w} & \mathbf{a}_{KIN,w} & \mathbf{\Theta}_{KIN,w} & \boldsymbol{\omega}_{KIN,w} & \lambda_{KIN} \end{bmatrix}^T \tag{4.11}$$

where $\mathbf{t}_{KIN,w}$ is the *unscaled* translation vector (and where now t_Z is estimated instead of $(t_Z\beta)$), $\delta\mathbf{v}_{KIN,w}$ is the full-scale incremental velocity vector, $\mathbf{a}_{KIN,w}$ is the full-scale acceleration vector, and λ_{KIN} is the scale factor, which is the quantity

$$\lambda = \frac{|\mathbf{t}_{w,abs}|}{|\mathbf{t}_{w,unscaled}|} \tag{4.12}$$

Note also that all quantities are in world frame coordinates, denoted by the subscript w.

Special consideration needs to be paid to the translation vector in Equation (4.11). The acceleration vector comes directly from the IMU and is a full-scale quantity. The same applies to the incremental velocity vector since this is derived by integrating the acceleration. However, in order to relate the translation vector in the kinematics EKF to the one from the SFM EKF it needs to transformed to an unscaled quantity. Thus, the time update equation for the unscaled translation vector must be re-written as:

$$\begin{aligned}\mathbf{t}_{KIN,w,k} &= \mathbf{t}_{KIN,w,k-1} + \frac{1}{\lambda_{KIN,k-1}}\left(\mathbf{v}_{KIN,w,k-2} + \delta\mathbf{v}_{KIN,w,k-1}\right)T + \frac{1}{2\lambda_{KIN,k-1}}\mathbf{a}_{KIN,w,k-1}T^2 \\ &= \mathbf{t}_{KIN,w,k-1} + \frac{1}{\lambda_{KIN,k-1}}\mathbf{v}_{KIN,w,k-1}T + \frac{1}{2\lambda_{KIN,w,k-1}}\mathbf{a}_{KIN,w,k-1}T^2 \end{aligned} \quad (4.13)$$

The full kinematics time update equation in matrix form is therefore given as:

$$\begin{bmatrix} \hat{\mathbf{t}}^-_{KIN,w,k} \\ \hat{\delta\mathbf{v}}^-_{KIN,w,k} \\ \hat{\mathbf{a}}^-_{KIN,w,k} \\ \hat{\mathbf{\Theta}}^-_{KIN,w,k} \\ \hat{\boldsymbol{\omega}}^-_{KIN,w,k} \\ \hat{\lambda}^-_{KIN,k} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & \frac{T}{\hat{\lambda}^+_{KIN,w,k-1}} diag\left(\frac{\hat{\mathbf{v}}^+_{KIN,w,k-2}}{\hat{\delta\mathbf{v}}^+_{KIN,w,k-1}} + 1\right) & \frac{T^2}{2\hat{\lambda}^+_{KIN,k-1}}\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & T\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & T\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{t}}^+_{KIN,w,k-1} \\ \hat{\delta\mathbf{v}}^+_{KIN,w,k-1} \\ \hat{\mathbf{a}}^+_{KIN,w,k-1} \\ \hat{\mathbf{\Theta}}^+_{KIN,w,k-1} \\ \hat{\boldsymbol{\omega}}^+_{KIN,w,k-1} \\ \hat{\lambda}^+_{KIN,k-1} \end{bmatrix} \quad (4.14)$$

where the set of elements shown in the 1st row, 2nd column involves element-wise division of two vectors. Therefore the prediction model Jacobian matrix is expressed as

$$J_{f,KIN} = \begin{bmatrix} \mathbf{I}_3 & \frac{T}{\hat{\lambda}^+_{KIN,k-1}}\mathbf{I}_3 & \frac{T^2}{2\hat{\lambda}^+_{KIN,k-1}}\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \frac{\partial f_1}{\partial\hat{\lambda}^+_{KIN,k-1}} \\ \mathbf{0}_3 & \mathbf{0}_3 & T\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & T\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1 \end{bmatrix} \quad (4.15)$$

where

$$\frac{\partial f_1}{\partial\hat{\lambda}^+_{KIN,k-1}} = \frac{1}{\hat{\lambda}^+_{KIN,k-1}}\hat{\mathbf{v}}^+_{KIN,w,k-2}T - \frac{1}{2}\hat{\mathbf{a}}^+_{KIN,w,k-1}T^2 \quad (4.16)$$

To fuse the vision and IMU data, two separate measurement update equations in the kinematics EKF are required: one to incorporate the estimates of un-scaled translation and total rotation output by the SFM EKF every time an SFM measurement update is carried out; and one to update estimates of the un-scaled translation and total rotation every time the IMU outputs measurements of instantaneous acceleration and rotational velocity. Therefore, there are two inputs to the kinematics EKF: (1) the results from the SFM EKF; and (2) the IMU. In the following equations the quantities output from SFM will be denoted by subscript V, and the quantities relating to the output from the IMU will

be denoted by the subscript I. The equations for the kinematics EKF IMU and vision measurement predictions are:

$$\begin{aligned}\hat{\mathbf{y}}_{KIN,I,k} &= \boldsymbol{H}_{KIN,I,k}\hat{\mathbf{X}}^{-}_{KIN,k} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1}\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1}\end{bmatrix}\hat{\mathbf{X}}^{-}_{KIN,k}\\ \hat{\mathbf{y}}_{KIN,V,k} &= \boldsymbol{H}_{KIN,V,k}\hat{\mathbf{X}}^{-}_{KIN,k} = \begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1}\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1}\end{bmatrix}\hat{\mathbf{X}}^{-}_{KIN,k}\end{aligned} \tag{4.17}$$

and the actual measurement vectors are expressed as

$$\begin{aligned}\mathbf{y}_{KIN,I,k} &= \begin{bmatrix}\mathbf{a}_{IMU,w,k}\\ \boldsymbol{\omega}_{IMU,w,k}\end{bmatrix}\\ \mathbf{y}_{KIN,V,k} &= \begin{bmatrix}\hat{\mathbf{t}}^{+}_{SFM,w,k}\\ \hat{\boldsymbol{\Theta}}^{+}_{SFM,w,k}\end{bmatrix}\end{aligned} \tag{4.18}$$

It is important to note that all quantities in Equation (4.18) are expressed in world frame coordinates, although they are not output from their respective sources in this form. A transformation must therefore be applied before inserting them into the kinematics EKF. The measurements returned from the IMU will be in the IMU coordinate system. Transforming these quantities into the world frame involves pre-multiplying by a rotation matrix describing the rotation from the IMU frame to the world frame. In practice this would involve applying a minimum of 2 rotations: one from the IMU frame to a body-fixed frame; and then another from the body-fixed frame to the world frame. Once in the world frame the gravity vector must be subtracted from the acceleration vector to give the pure translational acceleration. The body fixed frame in this work is the camera frame, and since the IMU is being simulated, a simplifying assumption is adopted that treats the IMU frame as being coincident with the camera frame. Therefore, the transformation from IMU frame to world frame involves pre-multiplying by the inverse of the current global rotation matrix, since the global rotation matrix describes the rotation from the world frame to the current camera frame. The global rotation matrix used in the SFM EKF describes the rotation from the world frame to the current camera frame, which is in the form required to obtain the total Euler angle rotation vector for the kinematics EKF, therefore no transformation is required.

Another consideration that must be taken into account is that in the SFM EKF the Z-component of the translation vector is estimated as a product of the inverse focal length and t_{Z_w} and so to incorporate this quantity as a measurement in the kinematics EKF it must be divided by the current estimate of the inverse focal length since in the kinematics EKF the Z-component is estimated directly.

4.6 Multi-Rate Sequence

In general, IMUs are capable of returning inertial measurements at a much higher rate than a camera is able to capture images. Therefore a final word relating to sequencing

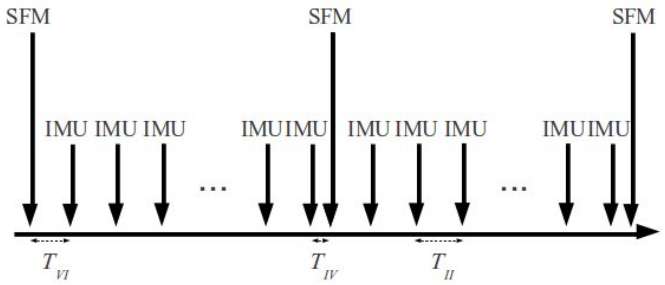


Figure 4.4: Multi-rate data sequence

is required. In this work it is assumed that the IMU returns measurements at 160Hz and the camera operates at a frame rate of 40fps. This would allow the IMU and camera to be perfectly synchronized, however in real-world systems this would be difficult to achieve, not least because a frame acquisition rate plus processing rate of 40fps would be unrealistically high. Therefore, while we presently ignore the unrealistic frame rate in order to reduce programming complexity, we do allow some flexibility with respect to the synchronisation by deliberately misaligning the two measurement sources as shown in Figure 4.4.

The two different updates are processed as soon as the measurements arrive, which results in a time distribution that may look like that shown in Figure 4.4. The time between an IMU update and a vision update is denoted by T_{IV}, the time between a vision update and an IMU update is denoted by T_{VI}, and the time between two consecutive IMU updates is denoted T_{II}. The appropriate value of T should be substituted into Equations (4.10), (4.13), (4.14), (4.15), and (4.16), depending on the situation.

4.7 Digital Elevation Model Construction

The unscaled structure parameters estimated in the SFM EKF can be used to generate a DEM of the scene visible in image 1 to enable hazard detection. In principle, this DEM could be generated at any point during the estimation process and iteratively refined in real-time as the motion, structure and scale factor parameters are updated in each step. However, currently, it is chosen to produce the DEM at the end of the estimation sequence once the parameters have been estimated to sufficient accuracy.

Due to the way that the system has been formulated, the structure parameters represent the un-scaled depth of the feature points along the Z-axis of the world frame, as shown in Figure 4.5. The first step in generating the DEM is to convert these values to fully scaled depths by multiplying by the estimated scale factor, i.e.

$$\hat{Z}_{w,abs,i} = \hat{\alpha}_{SFM,w,i}\hat{\lambda}_{KIN} \tag{4.19}$$

The corresponding $\hat{X}_{w,abs}$ and $\hat{Y}_{w,abs}$ coordinates for each feature point can then be

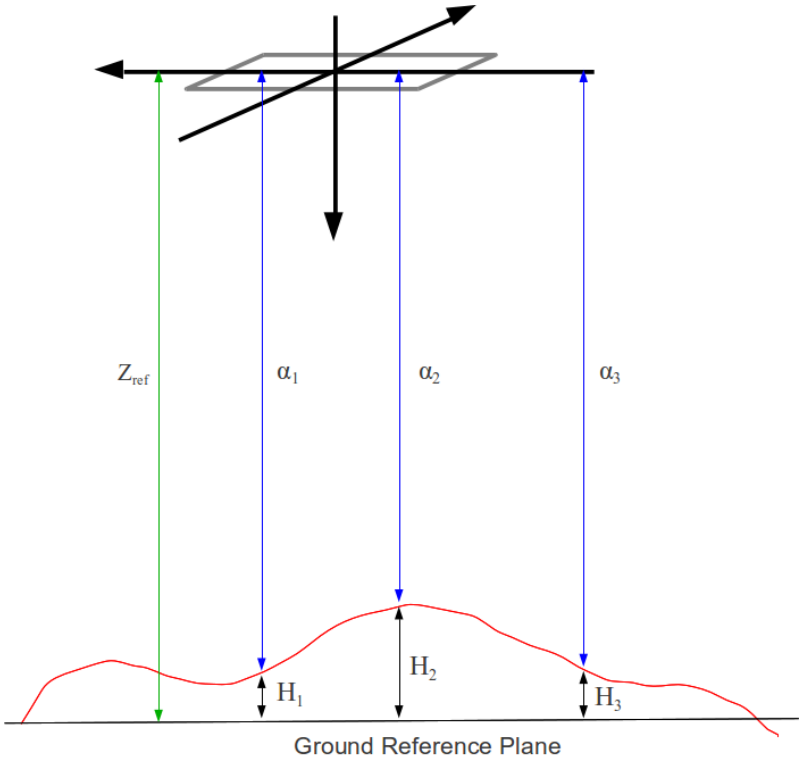


Figure 4.5: Relationship between parameters required for DEM construction relative to the world frame coordinate system

calculated using Equation (4.1) with $\hat{Z}_{w,abs}$, the estimated inverse focal length and the image plane coordinates in the first image.

Instead of depth in the world frame, the height of each point above a ground reference plane is required, as indicated in Figure 4.5. If the height of the world frame above the ground reference plane is known (i.e. the initial altitude at the start of the estimation process) then this is a simple case of subtracting the absolute depths from the height of the world frame, as follows:

$$\hat{H}_{abs,i} = Z_{ref} - \hat{Z}_{w,abs,i} \tag{4.20}$$

The resulting DEM will be relatively sparse, to an extent depending on the number of features remaining at the end of the estimation sequence. Thus to facilitate analysis, coarse 3D surfaces have been created by interpolating between the estimated points.

4.8 Results

To validate the developed algorithm it was first tested using pure synthetically generated data. That is, a set of regularly spaced artificial feature points with randomized depths

were generated and then their motion on the image plane was simulated (with added 5% random noise) for each image in a 100 frame sequence based on a straight forward trajectory. In this way the feature tracking stage has been skipped, which eliminates one source of potential errors. In order to model the image motion a camera with full field of view of $30°$ and a focal length of 30mm was assumed. The pure synthetic trajectory starts with the spacecraft at a height of 2000m, and moves straight down along the positive Z_w axis at a constant velocity of 100m/s, and rotating at a constant rate of 0.02rad/s anticlockwise about the Y_w axis to simulate swinging motion under a parachute. The IMU measurements were simulated based on an ADIS16385 6-degree of freedom inertial sensor by Analog Devices, which has a gyro noise spectral density of $0.04°\,\text{s}^{-1}\text{Hz}^{-1/2}$ and an accelerometer noise spectral density of $180 \times 10^{-6}\text{g Hz}^{-1/2}$. The sampling rate for the IMU was chosen to be 160Hz. Since it is assumed that the spacecraft is descending at terminal velocity (which is assumed to be known to high accuracy a priori), the acceleration measurements returned by the IMU are simulated to consist of just random noise. Likewise, the rotational velocity components are also random noise, apart from on the Y_w axis, which is simulated to be 0.02rad/s plus random noise. The results from the pure synthetic data are shown in Figure 4.6, along with their associated ground truth values.

Figure 4.6 a) and b) show fluctuations for the X- and Y-components of the absolute translation, respectively, starting with fluctuations as large as around 1m, but gradually decreasing to around 40–60cm near the end of the 100 frame image sequence to show better convergence. The fluctuating nature of these two results is to be expected since there is no significant motion in these parameters and the measurements from the IMU relating to these components consist only of random noise. Nevertheless, errors in position of a few tens of cm from a height of 2km are considered to be good performance and the observed decrease in fluctuation magnitude shows that these parameters may converge to greater accuracy over a longer sequence. The result shown in Figure 4.6 c) for the Z-component of absolute translation, on the other hand, is quite exceptional.

This is a significant result in the validation of the algorithm, because the Z-translation is arguably the most important component as it is the parameter in which there is the most significant motion in the chosen scenario, and it is also strongly related to the estimation of the structure parameters via Equation (4.12).

Figure 4.6 d) and f), for the X- and Z-components of the total rotation, also show fluctuations, again due to there being no significant motion in these parameters. However, it is important to note that the scale on these two graphs is 10^{-4}rad, meaning that at worst the magnitude of the fluctuations are less than $1/10$ of a degree. Figure 4.6 e) shows the estimation results for the Y-component of total rotation, and again these are very good since it is also a parameter where there is significant motion. Figure 4.6 g) shows very accurate results for the estimation of inverse focal length, which is another important parameter due to its relationship with the Z-component of translation. Since this is a very small number combined as a product with the relatively large Z-component of translation, slight errors in the inverse focal length can result in large errors in t_Z when the two values are separated. Figure 4.7 a) shows the ground truth DEM and b) the estimated DEM, which are presented in a top-down view, using identical colour scales to represent elevation. The importance of accurate motion estimation results is strongly

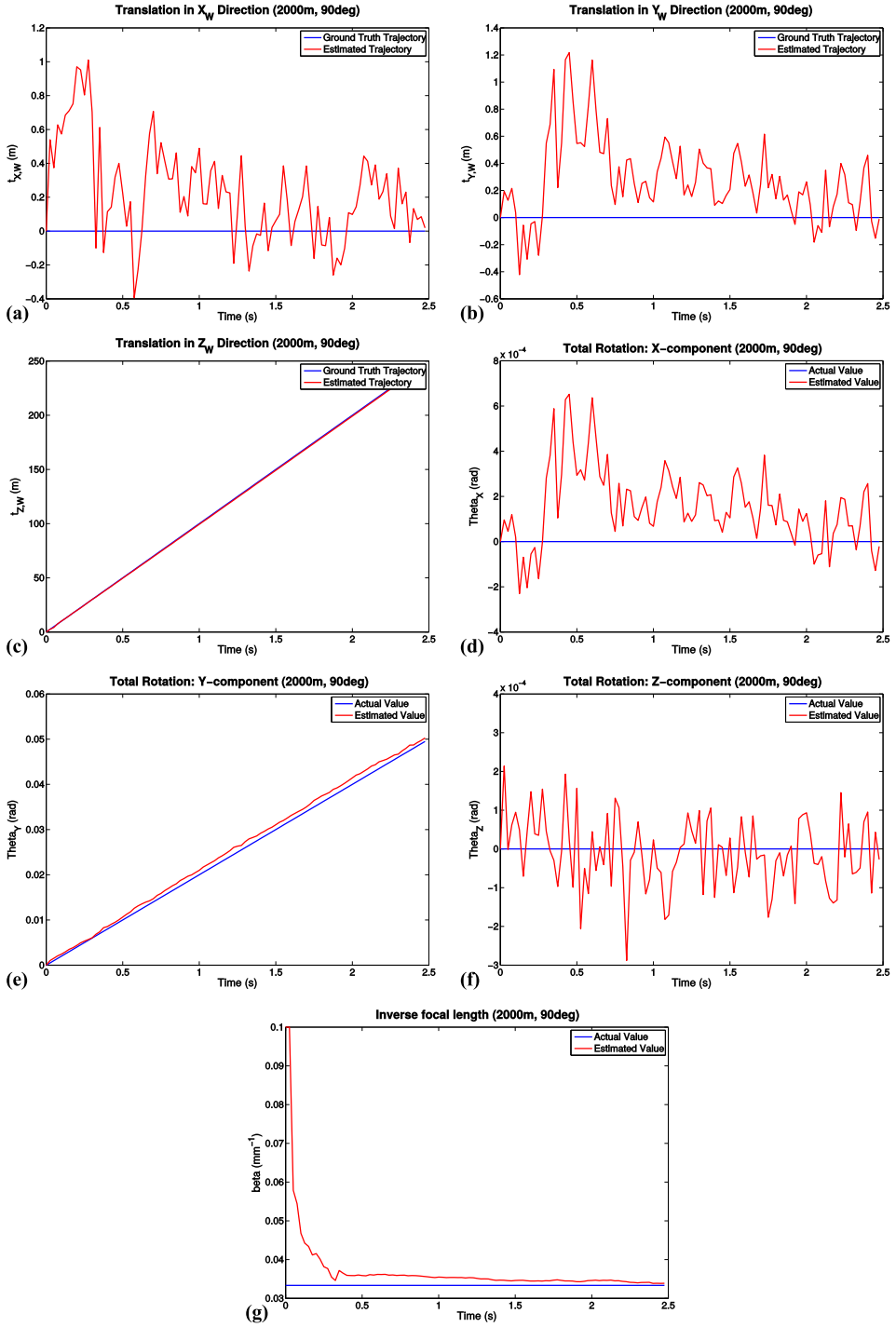


Figure 4.6: Motion results for pure synthetic data (2000m, $90°$ trajectory): (a) X-component of absolute translation; (b) Y-component of absolute translation; (c) Z-component of absolute translation; (d) X-component of total rotation; (e) Y-component of total rotation; (f) Z-component of total rotation; (g) inverse focal length

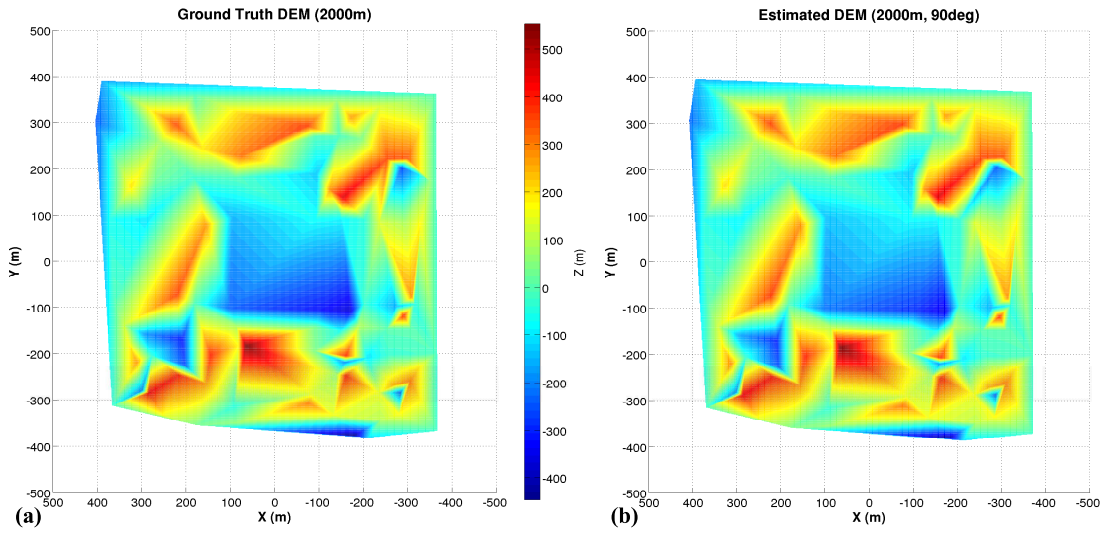


Figure 4.7: Structure results for pure synthetic data (2000m, $90°$ trajectory): (a) Top down view of ground truth DEM at equal sparsity to estimated DEM; (b) Top down view of estimated DEM

reflected in this figure, which shows that the DEM results are also exceptional as only very subtle differences can be seen between the two plots. The results obtained from the pure synthetic data experiment therefore provide a strong validation of the proposed algorithm.

A number of additional tests were carried out to further validate the performance of the algorithm, this time using actual images. These images were produced using the Planet and Asteroid Natural scene Generation Utility (PANGU), and represent physically realistic images of a spacecraft descending towards a remote planetary surface. DEMs of the terrain models used in these additional tests were created using the PANGU LIDAR tool in order to provide a dense ground truth of the surface structure. These DEMs are shown in Figure 4.8 for the three challenging test cases at altitudes of 2000m, 1000m, and 500m.

Testing with actual images requires the use of a method for detecting and tracking feature points through the entire image sequence. The feature tracking method adopted in this chapter is the conventional KLT algorithm. Feature detection methods are not perfect and so it is likely that a small number of features will be tracked erroneously. Therefore, these tests also provide insight into the general robustness of the algorithm, as well as providing a test of performance in a physically realistic scenario. In addition to this, a slight increase in the complexity of the motion has been introduced to provide a more realistic test trajectory. The effects of using different initial heights above the surface was also examined since this would result in larger inter-frame image motion, which may introduce instability in the results. Also, because the ultimate objective is to develop a system that will run continuously up until touchdown, this gives an indication of the performance at various altitudes. The first of these tests starts with a very challenging initial height of 2000m and has exactly the same motion parameters as in the

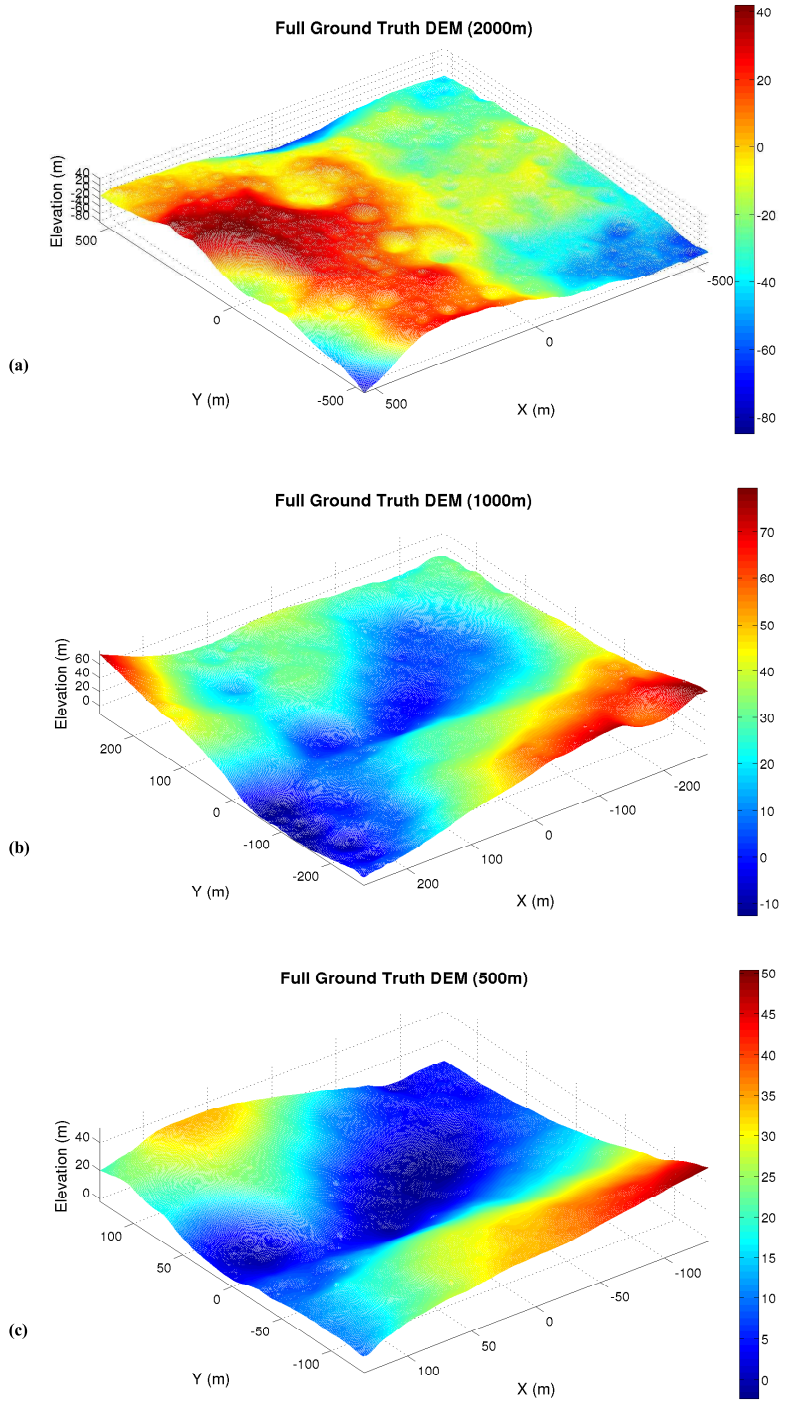


Figure 4.8: Full ground truth DEMs for PANGU surfaces used in tests at: a) 2000 m initial altitude; b) 1000 m initial altitude; c) 500m initial altitude

pure synthetic test in order to provide a direct comparison. Thus, it moves directly down at a velocity of 100m/s, with 0.02rad/s rotation on the Y_w axis. Again, the IMU measurements were modelled as random noise for the acceleration components since terminal velocity is assumed, and the rotational velocity measurements were simulated as random noise for the X-component and Z-component, and 0.02rad/s plus noise for the Y-component. The results for the motion parameters are shown in Figure 4.9, along with their corresponding ground truth values. Figure 4.8 a) shows a full dense ground truth DEM from a height of 2000m above the artificial surface. This DEM covers the same area as that in the field of view of the camera at a height of 2000m, based on a camera with $30°$ full field of view and focal length of 30mm. Figure 4.9 h) shows the estimated DEM alongside a sparse ground truth DEM, that was generated from the full ground truth DEM by selecting points that correspond to the features tracked throughout the image sequence. The estimation process was carried out over a 100 image sequence, as was done in the pure synthetic test.

The X-component of absolute translation in Figure 4.9 a) shows divergence by the end of the 100 frame image sequence, resulting in a final position error of around 10m. It was found that this result could be improved by a long iterative process of adjusting the tuning parameters in the EKF; however this had significant adverse effects on all the other motion parameters. It is important to note that this is a component in which there is no motion in the images so drift may be a consequence of the random noise in the IMU signal, or more likely due to outliers in the feature tracking process and errors due to linearisation in the EKF. Therefore further work into the cause and eradication of this is needed, such as outlier detection and removal and/or more robust filtering methods. The Y-component of translation, shown in Figure 4.9 b), shows fluctuations that appear along the image sequence. This is similar in behaviour to that observed in the pure synthetic test, indicating that the results may be driven by noise from the IMU. However, these fluctuations are considerably smaller in magnitude than those of the pure synthetic test (worst case 5 mm compared to 1m), thus these results can be considered to be very good. In Figure 4.9 c), remarkable performance is again observed in the estimation of the Z-component of translation, giving strong support for the potential of the algorithm for the same reasons discussed above.

In Figure 4.9 d), a considerable improvement in the results for the X-component of total rotation is observed compared to the results from the pure synthetic test, where now the scale on the Y-axis of the figure is 10^{-5}, compared to 10^{-4} previously. The rotation angle can be seen to be fluctuating at first, but quickly settles to much less severe fluctuations with a maximum peak-to-peak variation of around 4×10^{-5}rad, which equates to around $0.02°$. Therefore, it can be considered that this parameter is estimated to a high degree of accuracy, giving strong weight to the potential of this method for controlling the descent of a spacecraft to pin-point accuracy. Figure 4.9 e) shows some discrepancy between the ground truth value and the estimated value for the Y-component of total rotation. The results appear to follow the general trend, in that an approximately constant angular rate is estimated, but that this is slightly lower than the true value, meaning that the total rotation is underestimated by the end of the sequence. However the amount by which it deviates from the ground truth is only around 5.6×10^{-3}rad ($0.32°$) by the end of the sequence, which can still be considered a very good result. The results for the

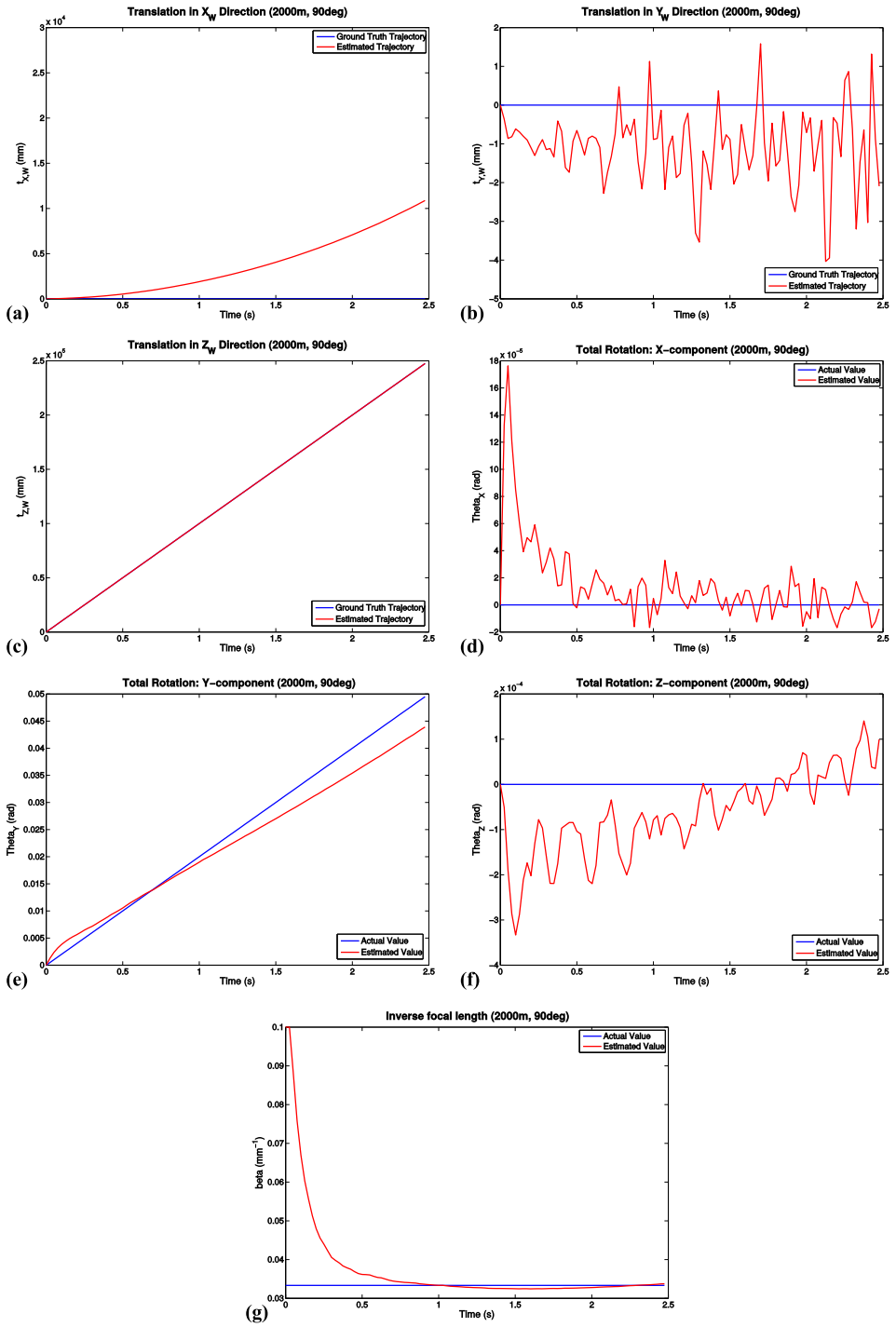


Figure 4.9: Motion results from 100 frame PANGU image sequence (2000m, $90°$ trajectory): a) X-component of absolute translation; b) Y-component of absolute translation; c) Z-component of absolute translation; d) X-component of total rotation; e) Y-component of total rotation; f) Z-component of total rotation; g) inverse focal length.

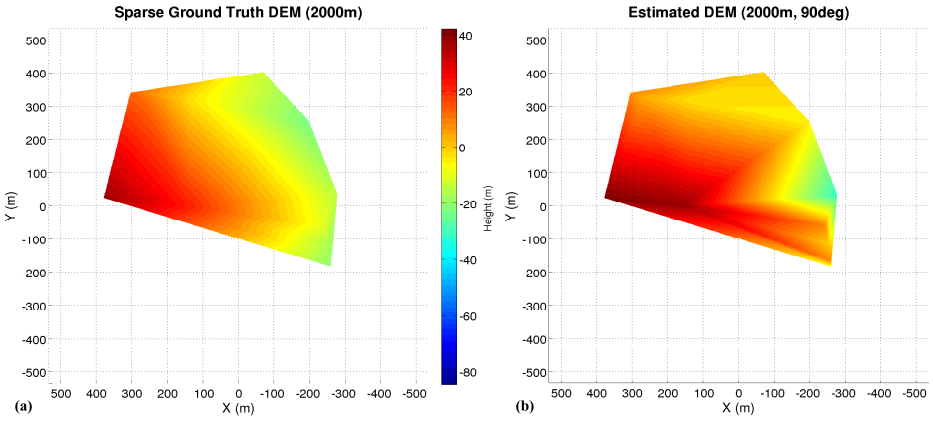


Figure 4.10: Structure results from 100 frame PANGU image sequence (2000m, $90°$ trajectory): a) Top down view of ground truth DEM of equal sparsity to estimated DEM; b) Top down view of estimated DEM.

Z-component of total rotation in Figure 4.9 f) are generally comparable to those obtained in the pure synthetic test. The rotation angle is seen to be fluctuating around the ground truth value, but in this case with slightly less severity. However, a slight positive trend is also observed that may or may not continue over a longer sequence. Regardless, this can still be considered a highly accurate estimation of the Z-rotation since in the worst case the estimation is in error by only $0.2°$ at the beginning of the sequence, and quickly settles to less severe fluctuations. Figure 4.9 g) also shows a very good estimation of the inverse focal length, which is seen to quickly converge to the ground truth value, finishing with a discrepancy of only 4.2×10^{-4}mm^{-1}.

Figure 4.10 shows that, for the most part, the estimated DEM (b) is in close agreement with the sparse ground truth DEM (a) over much of the region of coverage. However, it can be seen that there are still some anomalies around the edges, especially in the rightmost region. Over the estimated surface it can be seen that in the best-case there is a discrepancy of approximately only 3–5m, which is very acceptable considering this DEM is estimated from a height of 2km. However, in the worst-case there is a discrepancy of around 22m in the bottom left corner which is still reasonable, but shows that not all the feature points are estimated to high accuracy. The reason for this may be due to a combination of outliers in the feature tracking, EKF linearisation errors and errors in the estimation of the scale factor caused by the poorly estimated X-translation, which itself is likely due to the presence of outliers and linearisation errors. It is also worth mentioning that large numbers of features were lost during tracking over the 100 frame sequence, due mainly to divergence whilst iteratively refining the tracking solution in the KLT algorithm. Thus the resulting DEM is very sparse, and covers only a small area of the terrain in the field of view of the camera, as can be seen by the white region surrounding the DEM in

Figure 4.10, where there are no feature points present.

The results of a different test using PANGU images are now presented. This time the spacecraft starts at a height of 1000m, in order to provide some insight into the performance of the developed technique at a later stage of the descent, and also to investigate whether a better quality DEM can be obtained by starting at a lower altitude, as might be expected. In this case the velocity of the spacecraft is 80m/s straight down, which is assumed to remain constant throughout the 100 frame sequence. Rotational velocity of 0.02rad/s on the Z-axis is now also introduced to simulate some residual rotation from spin stabilization during the entry phase, as well as keeping the 0.02rad/s rotational velocity in the Y-axis. These results are shown in Figure 4.11.

Figure 4.11 a) again shows the presence of drift in the X-component of absolute translation, although now to slightly less severity, ending with a discrepancy of 7.8m. Figure 4.11 b) also seems to be showing signs of drift in the Y-component of absolute translation, although the final value is only around 33mm, so this can still be considered an exceptional result from the challenging height of 1km. Figure 4.11 c) again shows exceptional performance in the estimation of the Z-component of absolute translation, which is important as this is the parameter in which there is the most significant motion. Signs of drift are now also observed in the X-component of total rotation in Figure 4.11 d), but again the scale in the figure is 10^{-4}rad, meaning the final value is in error by only a fraction of degree, which is still a very positive result. Figure 4.11 e) shows similar general behaviour to that obtained previously in Figure 4.11 e) for the Y-component of total rotation. The final value in this case still only has a discrepancy of around 0.01rad ($0.6°$). Now that there is significant motion in the Z-component of rotation it is observed in Figure 4.11 f) that the results for this parameter are no longer driven by noise, and in fact it is observed that this component is estimated to a very high accuracy, where the final error is only $0.05°$. Figure 4.11 g) shows that again the results for the estimation of the inverse focal length are very good.

The degree of similarity between the estimated and sparse ground truth DEM shown in Figure 4.12 is now much greater. A few anomalies are still present; however, these are much less significant than in the previous case. The maximum discrepancy is now only around 8m, which is comparable to the best-case discrepancy in Figure 4.9. The best-case in Figure 4.12 is now only around 0.5m. It can be seen that the bottom portion of the DEM has roughly the same elevation values as is observed in the ground truth, thus indicating that the general shape and depth is correct for this region of the DEM. Overall the shape is very similar and the errors are much smaller, implying that greater accuracy can be expected as the spacecraft gets closer to the surface. However, comparing this DEM with that in the previous case, a clear difference in size can be observed. It was noticed that the rate of loss of feature points was much larger than in the case at 2000m, which was due to a greater tendency of divergence in the iterative tracking step of the KLT algorithm. This suggests that as the camera gets closer to the surface and the inter-frame image motion increases, there is significantly reduced stability in the KLT tracker. This results in a much sparser DEM, and in this case resulted in a much smaller DEM due to the remaining features being relatively closely clustered.

Figure 4.13 presents the results from another PANGU sequence, with the spacecraft starting at an initial altitude of 500m, to investigate whether decreasing the altitude even

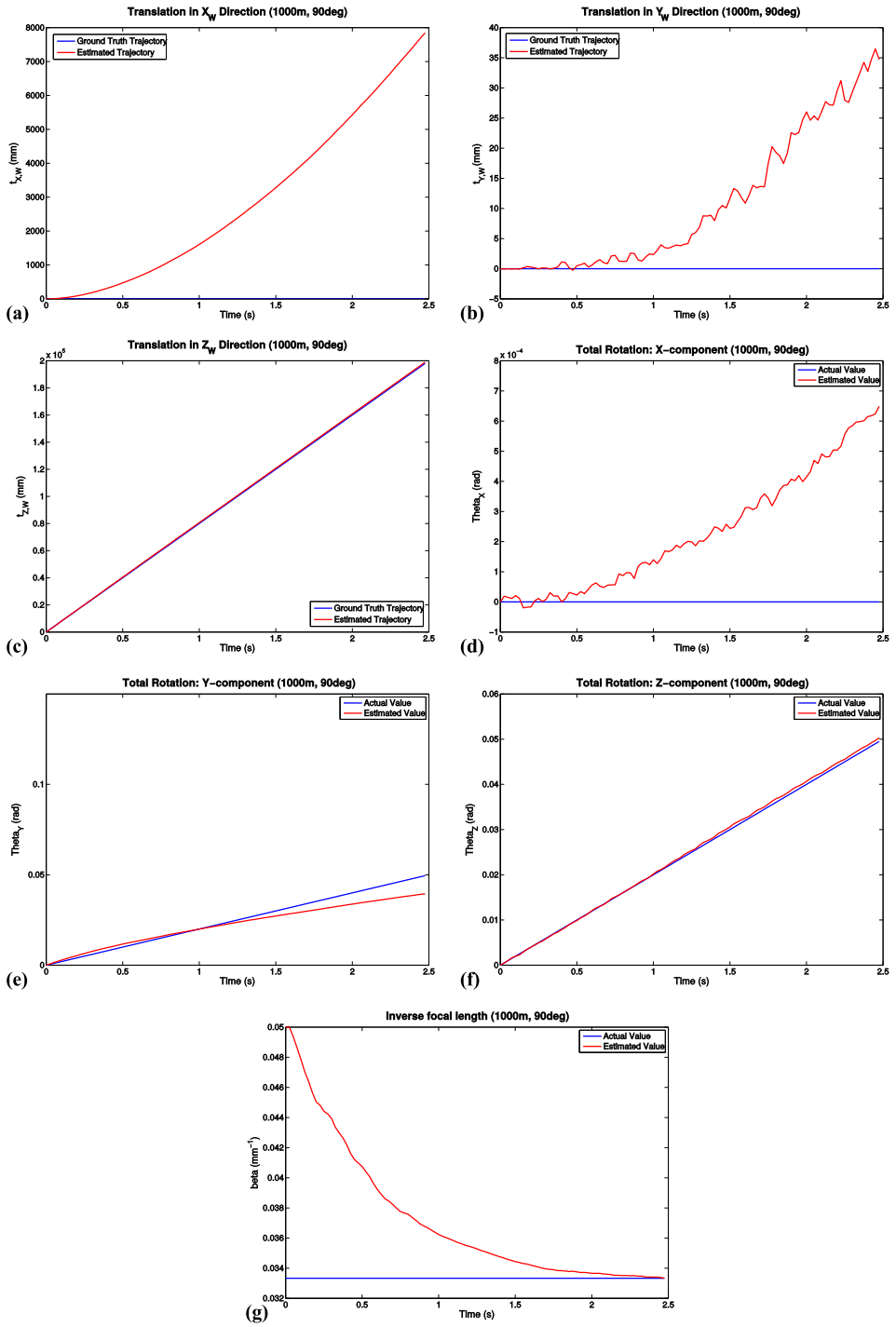


Figure 4.11: Motion results from 100 frame PANGU image sequence (1000m, $90°$ trajectory): a) X-component of absolute translation; b) Y-component of absolute translation; c) Z-component of absolute translation; d) X-component of total rotation; e) Y-component of total rotation; f) Z-component of total rotation; g) inverse focal length.

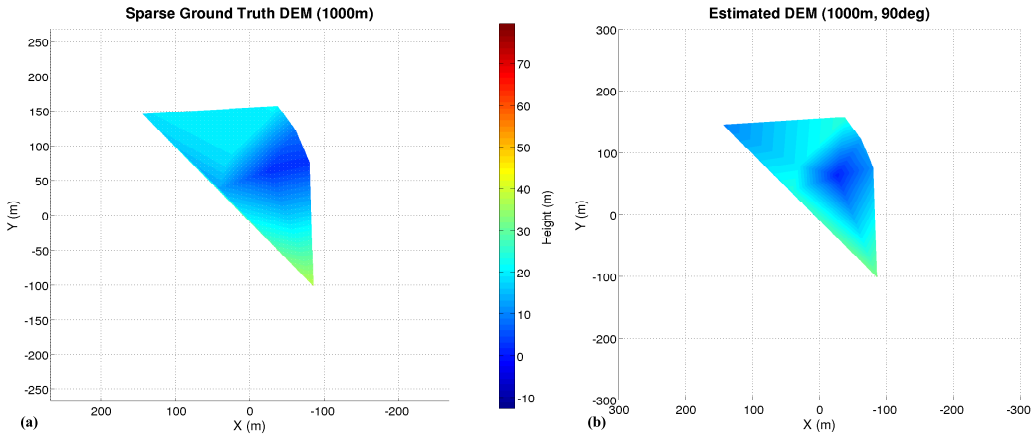


Figure 4.12: Structure results from 100 frame PANGU image sequence (1000m, $90°$ trajectory): a) Top down view of ground truth DEM with equal sparsity to estimated DEM; b) Top down view of estimated DEM.

further will lead to further improvement in the DEM as was seen above when going from 2000m to 1000m. In this case the velocity of the spacecraft is 60m/s, and again the trajectory was chosen so that the spacecraft is moving directly downwards. The remaining motion parameters are identical to those in the 1000m case.

With the initial height at 500m there is a further improvement in the X-component of translation in Figure 4.13 a), although drift is still observed in this parameter. The final value is now in error by around 5m. If nothing else this supports the hypothesis that the results are expected to improve as the spacecraft gets closer to the surface. The Y-component of translation also shows an improvement over the results obtained for the 1000m case, where it is observed in Figure 4.13 b) that the worst case error is around 12mm. Figure 4.13 c) again shows very good results for the Z-component of translation, as in all the previous cases. Figure 4.13 d) shows a marginal improvement in the X-component of total rotation and the final discrepancy in this parameter is still only a fraction of a degree. In Figure 4.13 e) a slight degradation in the estimation of the Y-component of total rotation is observed compared to the 1000m case. This is likely due to a slight increase in the number of feature outliers or increased linearisation error as the inter-frame image motion increases as the spacecraft gets closer to the surface. The results in Figure 4.13 f) and g) for the Z-rotation and inverse focal length, respectively, show that these two parameters are still estimated to a very high degree of accuracy, as was observed in the 1000m case. In this case the Z-rotation can be seen to be fractionally better than in the previous case, and the inverse focal length appears to converge quicker.

Figure 4.14, in general, shows a high degree of similarity between the two DEMs, with the overall shape appearing to agree quite closely. Therefore we can conclude that this shows an improvement over the DEMs presented in Figure 4.12, although the

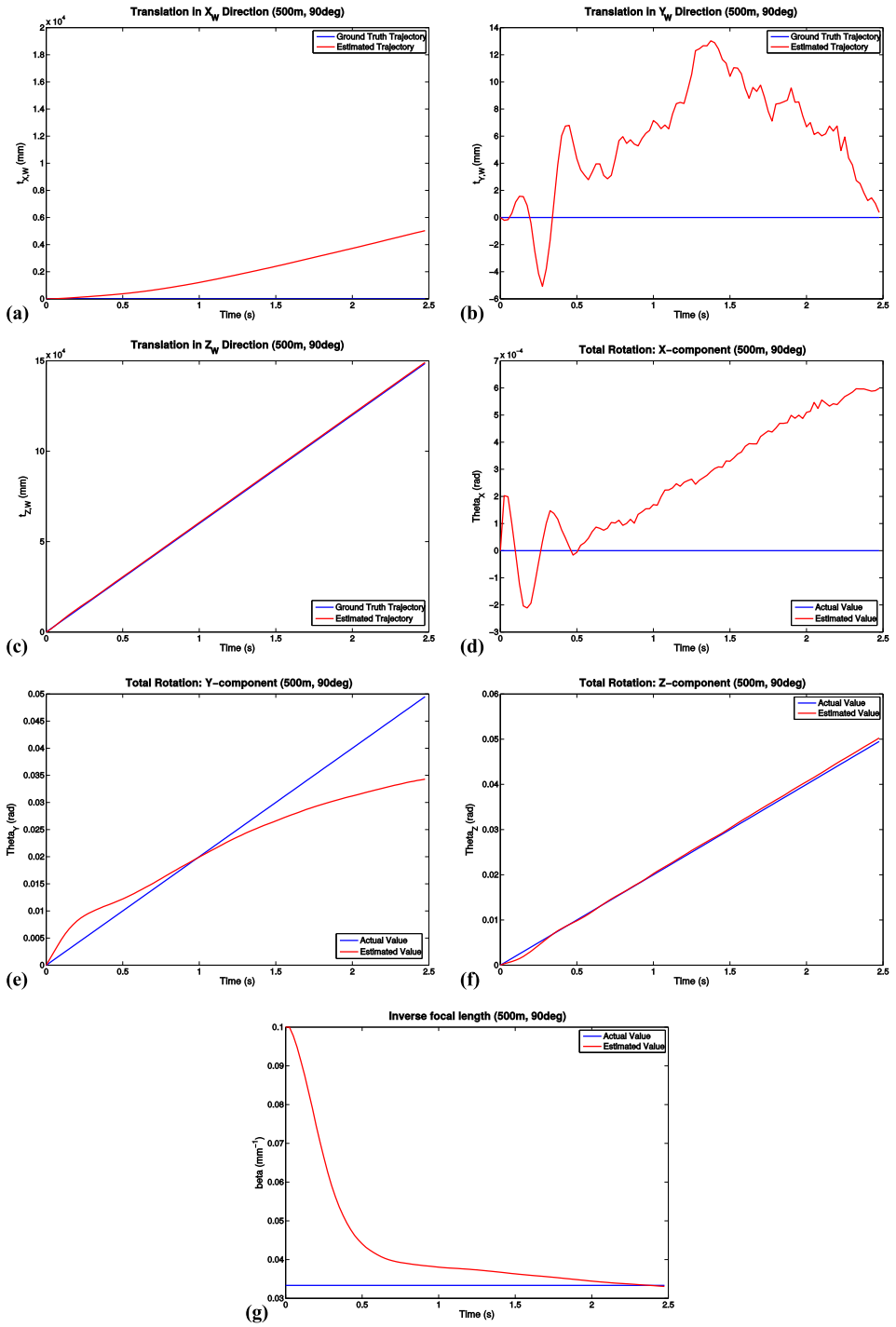


Figure 4.13: Motion results from 100 frame PANGU image sequence (500m, $90°$ trajectory): a) X-component of absolute translation; b) Y-component of absolute translation; c) Z-component of absolute translation; d) X-component of total rotation; e) Y-component of total rotation; f) Z-component of total rotation; g) inverse focal length.

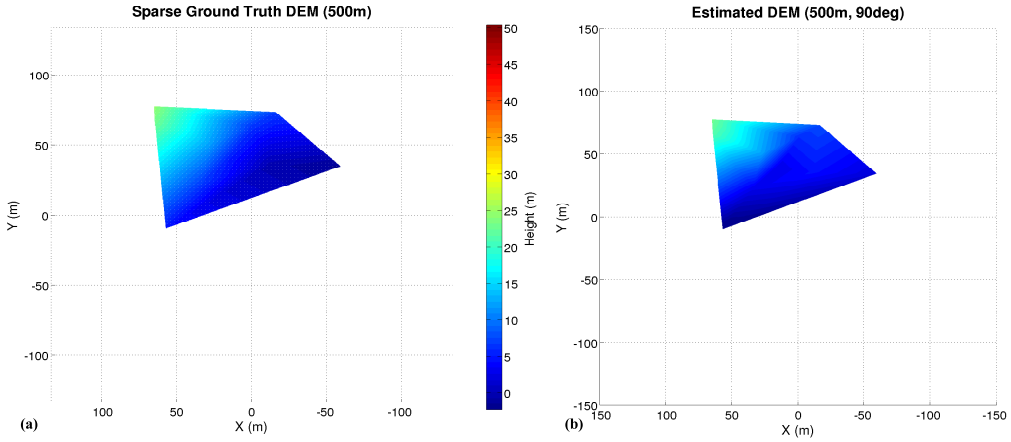


Figure 4.14: Structure results from 100 frame PANGU image sequence (500m, $90°$ trajectory): a) Top down view of ground truth DEM with equal sparsity to estimated DEM; b) Top down view of Estimated DEM.

improvement is not as dramatic as that observed between Figure 4.10 and 4.12. The best case discrepancy is now 0.4m (compared with 0.5m previously) and the worst case is now approximately 6m (compared with 8 m previously). Therefore, these results give further support to the expectation that more accurate results should be obtained as the spacecraft gets nearer to the surface.

The above DEMs, particularly those of Figure 4.12 and 4.14, are particularly small and sparse. This is a consequence of a large number of feature points being lost during the tracking process. The main reason for the loss of features may be due to drift in the minimization step of the KLT algorithm, which is based on a pure translation image motion model (see [29, 46–48] for details). If the tracking solution for a particular feature does not converge after a fixed number of iterations, divergence is assumed and that feature is rejected. This explains the sparsity, and to an extent the size of the resulting DEM, however the actual size of the DEM is coincidental since it just so happened that the remaining features were relatively closely clustered in Figure 4.12 and 4.14.

To examine the potential of the current method for producing accurate DEMs of a larger area, and to increase the density of the DEM, the number of images in the sequence was reduced so as to retain a much larger number of features spanning a larger area of the initial image. Figure 4.15 presents the results from the trajectory with 500m initial altitude, with velocity of 60m/s in the Z_w axis as before, but now for only a 36 frame image sequence.

The results for the motion parameters in Figure 4.15 in general show a slight improvement in all parameters except the Y-component of translation in which some drift is observed although this is very slight as it only drifts to a value of 8 mm. Also in the X-component of translation, a significant improvement is seen (final value of 0.8m

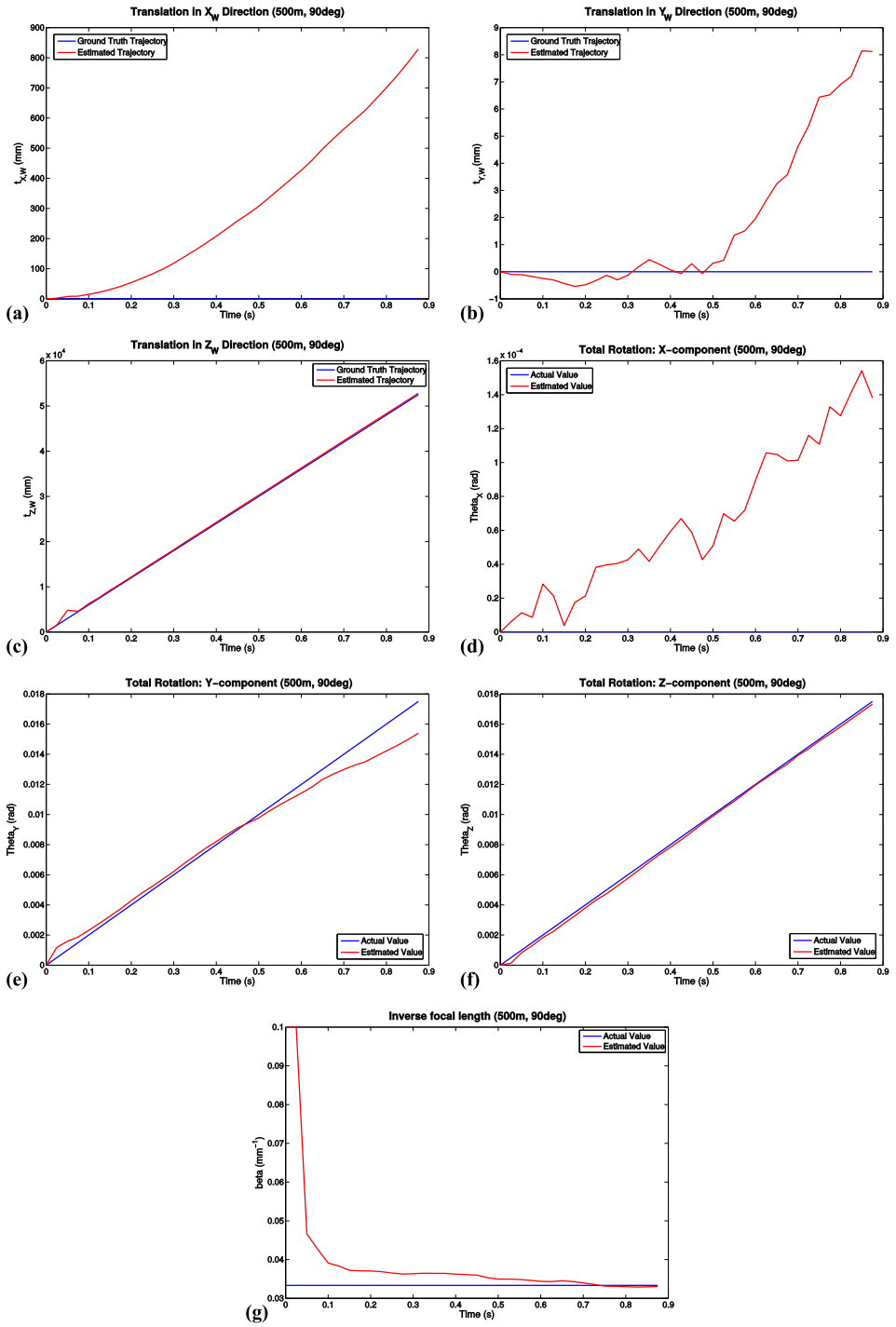


Figure 4.15: Motion results from 36 frame PANGU image sequence (500m, $90°$ trajectory): a) X-component of absolute translation; b) Y-component of absolute translation; c) Z-component of absolute translation; d) X-component of total rotation; e) Y-component of total rotation; f) Z-component of total rotation; g) inverse focal length.

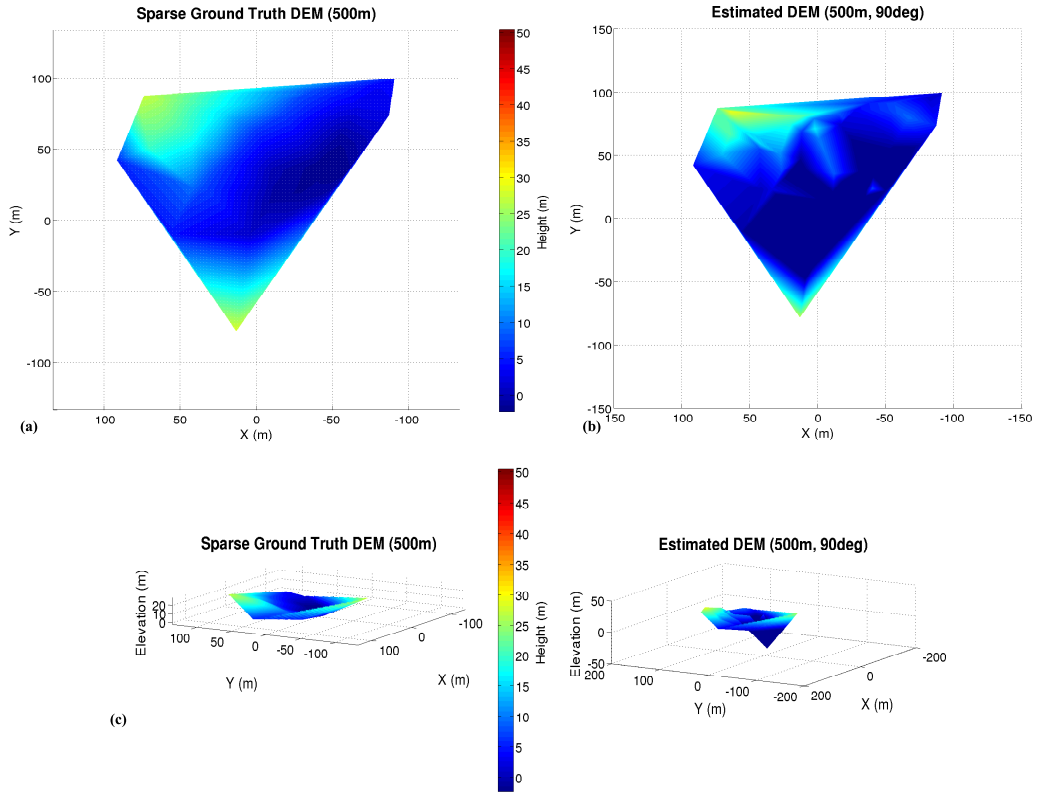


Figure 4.16: Structure results from 36 frame PANGU image sequence (500m, $90°$ trajectory): a) Top down view of ground truth DEM with equl sparsity to estimated DEM; b) Top down view of estimated DEM; c) Rotated view of ground truth and estimated DEMs.

compared to 5m) even though the drift behaviour is still present. The DEM in The DEMs in Figure 4.16 are now around twice as large and can be seen to be denser from the fact that some clear fluctuation is now observed in the surface. However this fluctuation should not be occurring as can be seen by comparing it with the ground truth DEM. Overall the estimated depth/elevation values appear to be in relatively close agreement over much of the surface. However it can be seen that now the minimum error is approximately 2m and the maximum error is around 22m. The point at which the worst case error occurs can be clearly seen in Figure 4.15 i), which shows the DEM from a different view point. This point is a clear outlier in the DEM, which otherwise is showing good agreement for the overall shape of the estimated terrain. This again highlights the need for a reliable method of removing outliers from the feature tracking or more robust filtering techniques. The fact that the DEM has reduced general elevation accuracy may also suggest that the depth parameters require a longer sequence in order to converge. Further work is required to determine if this is the case, and if so a modified feature tracking method may need to be employed to enable the features to be tracked reliably

for longer period of time. However, it can be concluded from the results of the motion estimation that the use of a greater number of image features has the potential to lead to a greater degree of accuracy in the estimated parameters.

4.9 Conclusions

In this chapter a multi-source, multi-rate data fusion algorithm has been presented that combines recursive feature-based structure from motion with measurements from an inertial measurement unit, as a means to directly estimate the unknown scale factor present in single camera structure from motion algorithms. It has been demonstrated that this approach is capable of providing highly accurate and robust of estimates of the 3D motion parameters and scene structure using a single camera. It is believed that the approach proposed has very strong potential to supply the sensing accuracy required to achieve the landing precision and hazard detection requirements of future missions to autonomously land a spacecraft on the surface of a remote planetary body at a preselected site of scientific interest.

The work presented here represents only a small part of what would be required in a full, next-generation entry, descent and landing system, and thus only tackles a subset of the capabilities needed by such a system. However, it provides an important first step and solid foundation on which to build the required extra capabilities. Although very promising results have been shown for the estimation of the motion parameters and DEM construction, a further investigation to provide an alternative feature detection algorithm and a more robust filtering technique is possible. The issue of sparsity must be addressed in the currently estimated DEM in order to maximize its usefulness in hazard detection and avoidance. Therefore, many more features, and the ability to track them for longer periods of time is clearly required. Alternatively, an altogether different method could be applied for estimating the structure of the terrain, such as a shape from shading algorithm, which could then be combined with the current technique as a way of filling in the gaps between the feature points.

5 INCREASING ROBUSTNESS

The structure from motion framework described in Chapter 4 was chosen due to its reported claims of increased robustness compared to the other techniques discussed in the literature. The same is also true for the decision to adopt a feature-based method over an optical flow-based method. In this chapter, we look at additional measures that can be employed to further increase the robustness of the estimation of the scene structure by examining ways in which the filtering framework can be made more robust in comparison to the conventional techniques such as the standard extended Kalman filter, as used in the previous chapter. In this chapter, we present two different alternative filtering approaches that have been applied in an attempt to improve upon the results presented in Chapter 4. The first of these is a method based on the H_∞ filter, which despite its potential for greatly improving robustness, unfortunately did not result in particularly significant improvements. The second approach is by far the largest contribution of this thesis, which consists of a fully adaptive, self-initialising filtering framework based upon a parallel master-slave filtering framework built on the square-root unscented Kalman filter. In this filtering arrangement the master filter is responsible for estimating the structure parameters and the slave filter is responsible for adaptively re-tuning the master filter in the event of suboptimal initial tuning parameters and possibly changing noise statistics. The chapter concludes with a set of proof-of-concept structure estimation results that demonstrates the feasibility of the approach. More conclusive testing of this method is presented in the remaining chapters of this thesis.

5.1 State Estimation Algorithms

The most widely used filtering method in state estimation for linear systems is the Kalman Filter (KF). The reason for this is that it is relatively simple to implement, it is robust, tractable and it is provably optimal in the case of Gaussian noise distributions for both the measurement and process noise. However, many real life systems are not linear in nature, therefore the application of the Kalman filter to non-linear systems can be very difficult. One of the most common approaches is to implement the Extended Kalman Filter (EKF), which linearises the process and measurement model equations about the current state estimate, via the use of a first order Taylor expansion, so that the linear KF equations may be used to update the state of the system. This linearisation step gives rise to a filtering framework that is sub-optimal, which in some cases can be a source of significant error, especially when the system is highly non-linear (i.e. when the higher order terms of the Taylor expansion are significant), in which case the performance of the EKF can become severely degraded, often to the point where divergence may occur [82]. The EKF can, therefore, often only be reliably applied in situations where the system dynamics are almost linear on the time scale of the update

intervals [86]. Another potential drawback from standard Kalman filtering theory is the underlying assumption that the noise processes acting on the system are assumed to be fully known and governed by a Gaussian distribution, as mentioned above. However, this assumption can often be invalid in many applications, leading to further difficulties in successfully applying the EKF.

These issues have been well known for a long time, and so many attempts have been made to derive alternative filtering techniques that either attempt to entirely bypass those aspects of the EKF that lead to the above mentioned weaknesses, or that attempt to overcome these limitations by incorporating additional mechanisms or constraints in order to increase the overall robustness of the filter, despite the inherent limitations, to prevent the filter from diverging. An example of a filtering technique that falls into the latter of these two categories is the extended H_∞ filtering ($EH_\infty F$) method developed by Einicke and White [87, 88]. The H_∞ filter achieves robustness in the face of uncertainty in the system model and/or uncertainty in the characteristics of the noise disturbances by effectively minimising the worst-case estimation error. In Section 5.2 we present a detailed discussion of the $EH_\infty F$ and assess its suitability for SFM estimation in the context of planetary descent. Despite performing well in motion estimation, it unfortunately was found not to offer any significant improvement in the estimation of the structure parameters, thus leading to the development of an alternative method that falls into the first category mentioned above. This second approach is the subject of Section 5.6, but some additional details are also presented in the remainder of this section.

To overcome the limitations of the EKF when applied to significantly non-linear systems, Julier and Uhlmann [86, 89], and Julier, Uhlmann and Durrant-Whyte [90] proposed and demonstrated the effectiveness of a Kalman filter type algorithm that does not require linearisation of the system models. This is achieved through the use of the unscented transform, which is based on the idea that it is easier to approximate a probability distribution than it is to approximate an arbitrary non-linear function [86, 89, 90]. In the unscented transform, a small number of deterministically chosen points are selected such that their sample mean and covariance match the distribution that is being approximated. These points, known as sigma points, are then transformed using the full non-linear system models, to yield a cloud of transformed points from which an estimate of the transformed mean and covariance can be calculated. This method of propagating the probability distribution of the state of the system was demonstrated to produce much more statistically consistent estimates of the true distribution than can generally be achieved using the EKF. The unscented transform has the added benefit of being able to approximate non-Gaussian distributions since its deterministically chosen sigma points can encode higher order information. The resulting filtering algorithm, known as the Unscented Kalman Filter (UKF), is, according to [89], also of the same order of computational complexity as the EKF, therefore it does not require any additional computational resources to implement. For these reasons, the authors of [86, 89, 90] recommend the use of the UKF algorithm over the EKF in virtually all applications.

Despite the reported benefits of using the UKF over the EKF, it is not without its implementation difficulties. Covariance matrices are, by definition, at least positive semi-

definite matrices. In the conventional UKF algorithm, calculation of the sigma points in each time step requires taking the square-root of the state covariance matrix, which is only uniquely defined for positive definite or positive semi-definite matrices. For any other type of matrix a solution is not possible. However, due to machine inaccuracies in representing floating point numbers, the repeated matrix square root operation, often achieved using the Cholesky factorisation, can lead to an accumulation of errors that can result in the state covariance matrix becoming non-positive semi-definite, causing the Cholesky factorisation to fail. For this reason, and also for the fact that the Cholesky factorisation is the most computationally expensive operation in the UKF algorithm, van der Merwe and Wan [91] proposed the Square-Root Unscented Kalman Filter (SR-UKF), which directly propagates the square root of the state covariance matrix instead of the full state covariance in order to avoid the need to re-factorise for the computation of the sigma points in each time step. The resulting algorithm is not only more computationally efficient (in certain formulations), but it also guarantees positive semi-definiteness of the state covariance matrix [91], which may lead to improved performance and increased ease of implementation. It is for this reason that the SR-UKF was selected as the backbone filtering framework for the work presented in Section 5.6 of this chapter.

5.2 H_∞ Filtering

The majority of recursive SFM algorithms presented in the literature utilise estimator frameworks based on the L_2 norm, mainly in the form of the extended Kalman filter (EKF). However, as briefly touched upon above, it is widely known that the EKF has a number of limitations that may lead to significant degradation of the accuracy and robustness of the computed solution if certain conditions relating to the underlying assumptions of the EKF are not met. The EKF assumes that the statistics of the noise signals are known in advance and that this noise is governed by a Gaussian distribution. This is often not true in practice, and as such may lead to instability in the filtered results. The EKF is used when either the measurement or dynamics model, or both, are non-linear functions of the state variables. In each time-step, a linearisation about the current state parameters is carried out via the use of a first order Taylor expansion, and then the state is updated using the linear Kalman filter update equations. This linearisation may be a source of significant error, especially when the system is highly non-linear (in this case the higher order terms of the Taylor expansion become significant), in which case the performance can become severely degraded [82].

The EKF attempts to minimise the variance in the estimated state variables, but this is a sub-optimal solution when the measurement and dynamics models are non-linear functions. A more robust filtering framework, known as H_∞ filtering, can be derived through an application of game theory in which nature (the adversary) is viewed as trying to maximise the error in the state by introducing large disturbances in the system, whereas the engineer's goal is to try and find a state estimate that minimises the error. By setting up a cost function in which the numerator consists of the weighted mean square error of the state (the property that we are trying to minimise) and the denominator consists of the disturbances that nature is trying to maximise, this forces

nature to play fair by ensuring the disturbances remain bounded. This type of cost function is difficult to minimise directly so instead a performance bound is applied and an estimation strategy is developed to ensure that a state estimate is found that results in the cost function being less than the chosen upper bound. In other words the goal is to find estimates that satisfy a worst-case performance criterion. For more details see [87, 88, 92, 93]. This type of estimation strategy is more robust in the face of unknown noise statistics and modelling uncertainty, and can be easily extended to allow higher order terms in the Taylor expansion to be modelled as additional noise uncertainties that are functions of the state estimation error, thereby also making it more robust when dealing with highly non-linear systems [82]. The EH$_\infty$F approach of [87, 88] was adopted for the filtering framework of a similar recursive SFM algorithm for use on an unmanned aerial vehicle (UAV) equipped with a single camera by other researchers in our department, where it was found that it offered superior performance over the classic EKF. The results of this work are presented in [82]. It has also been applied in this project, and published in a conference paper – see [94], where it was found to offer an improvement in the estimation of the motion parameters compared to the EKF. However, it was unfortunately found not to offer any improvement in the estimation of the structure parameters, and so an alternative method was ultimately sought that will be presented in Section 5.6. However, until then we proceed with a presentation of an H$_\infty$ method similar to that of Einicke and White [87, 88] applied to the highly non-linear SFM algorithm presented in the previous chapter, as an attempt to improve estimation accuracy and robustness.

Consider a system with a non-linear measurement function

$$\mathbf{y}_k = h\left(\mathbf{x}_k, \mathbf{v_k}\right) \quad , \tag{5.1}$$

where $\mathbf{x}_k$ is the state vector and $\mathbf{v}_k$ is the measurement noise vector. Expanding h in a Taylor series about the current state estimate $\hat{\mathbf{x}}_k$ gives

$$h\left(\mathbf{x}_k\right) + \boldsymbol{H}_{\hat{\mathbf{x}}_k} \cdot \left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) + \Delta_h\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) \quad , \tag{5.2}$$

where Δ_h represents the higher order terms of the Taylor series expansion, which is a function of $\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)$, and $\boldsymbol{H}_{\hat{\mathbf{x}}_k}$ is the Jacobian matrix of h with respect to the state variables and evaluated at $\hat{\mathbf{x}}_k$. Rearranging Equation (5.2) gives a linearised equation for the measurement function

$$\mathbf{z}_k = \boldsymbol{H}_{\hat{\mathbf{x}}_k}\mathbf{x}_k + \mathbf{v}_k + \mathbf{q}_k + \mathbf{r}_k \tag{5.3}$$

where $\mathbf{q}_k = h\left(\hat{\mathbf{x}}_k\right) - \boldsymbol{H}_{\hat{\mathbf{x}}_k}\hat{\mathbf{x}}_k$ and $\mathbf{r}_k = \Delta_h\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)$, which expresses the uncertainty as an additional exogenous input satisfying the H_∞ performance criterion $\left\|\mathbf{r}_k\right\|_2^2 \leq \delta_h^2 \left\|\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)\right\|_2^2$.

Uncertainties due to modelling errors in the system equation can be incorporated into the model in a similar way to which the higher order terms of the measurement model were included. This leads to a process model equation of the form

$$\mathbf{x}_k = \left(\boldsymbol{F}_{\hat{\mathbf{x}}_{k-1}} + \Delta_F\right)\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{5.4}$$

where $\boldsymbol{F}_{\hat{\mathbf{x}}_{k-1}}$ is the process model Jacobian matrix, Δ_F represents the higher order terms in the Taylor series expansion of the non-linear process model equation, and $\mathbf{w}_k$ is the process noise vector. Rearranging Equation (5.4), and assuming that $\mathbf{s}_k = (\Delta_F\, \mathbf{x}_{k-1})$ is proportional to $\Delta_F\,(\mathbf{x}_k - \hat{\mathbf{x}}_k)$, the uncertainty of the system matrix can be associated with the estimation error. This results in

$$\mathbf{x}_k = \boldsymbol{F}_{\hat{\mathbf{x}}_{k-1}}\, \mathbf{x}_{k-1} + \mathbf{w}_{k-1} + \mathbf{s}_k \tag{5.5}$$

where $\mathbf{s}_k$ represents an additional exogenous input satisfying the H_∞ performance criterion $\|\mathbf{s}_k\|_2^2 \leq \delta_F^2\, \|(\mathbf{x}_k - \hat{\mathbf{x}}_k)\|_2^2$.

The system represented by Equations (5.3) and (5.5) can be solved by introducing a scaling of the input noise signals in lieu of the additional inputs $\mathbf{s}_k$ and $\mathbf{r}_k$, which can be regarded as simple weights. The problem may then be re-written as

$$\begin{aligned} \mathbf{x}_k &= \boldsymbol{F}_{\hat{\mathbf{x}}_{k-1}} \mathbf{x}_{k-1} + c\mathbf{w}_{k-1} \\ \mathbf{z}_k &= \boldsymbol{H}_{\hat{\mathbf{x}}_k} \mathbf{x}_k + c\mathbf{v}_k \end{aligned} \tag{5.6}$$

where $c = 1 - \gamma^2\delta_{h2} - \gamma^2\delta_F^2$. See [88] for proof of this approach and further details.

The EH$_\infty$F can now be obtained by applying the linear H$_\infty$ equations in a similar way to which the EKF is obtained using the linear Kalman filtering equations. This results in a set of equations that are identical to the standard EKF equations in all but the state covariance update equation in the measurement update step, which is now expressed as

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{P}_{k|k-1} \begin{bmatrix} -\boldsymbol{I} & \boldsymbol{H}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_{k|k-1} - \gamma^2\boldsymbol{I} & -\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T \\ -\boldsymbol{H}_k\boldsymbol{P}_{k|k-1} & \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k \end{bmatrix}^{-1} \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{H}_k \end{bmatrix} \boldsymbol{P}_{k|k-1} \tag{5.7}$$

Equation (5.7) represents the only change to the formulation of the SFM algorithm that was presented in Chapter 4, except that this now introduces an additional tuning parameter, γ, that must be manually tuned for optimum performance. In addition to this, we now also investigate an extra means of increasing robustness, not by making further changes to the SFM filtering formulation, but by making use of a modified approach to KLT tracking that aims to increase robustness through the introduction of information provided by inertial measurements. This modification to the KLT tracking algorithm was discussed in Chapter 3.

5.3 H_∞ Results

Testing the performance of the EH$_\infty$F with IMU-KLT was carried out in much the same way as the tests in Chapter 4 in that a sequence of 100 artificial images were generated from a simulated planetary surface using PANGU. The image sequence represents a short section of the descent phase for a spacecraft following a semi-realistic trajectory. The results presented in this section are again of a deliberately challenging scenario in

which the spacecraft begins at an initial height of 2000m above the surface. The initial velocity of the spacecraft was chosen to be 100m/s straight down, again roughly based on the expected velocity profile for the Mars Science Laboratory spacecraft, presented in [85], in order to provide representative test conditions at this altitude. In addition to this, a rotational velocity of 0.02rad/s was applied to both the Y- and Z-axes to simulate pendulous motion under a parachute and some residual spin from spin-stabilisation during the entry phase. Both the translational and rotational velocities were assumed to remain constant over the entire 100 image sequence. All other motion parameters were set to zero with random noise applied for the IMU measurements using representative noise figures based on an ADIS16385 6-degree of freedom inertial sensor by Analog Devices. Figure 5.1 presents the results for the estimated motion parameters along with ground truth values.

Figure 5.1 (a) presents the results for the X-component of full scaled translation of the spacecraft with respect to the world frame. This graph indicates that there is an issue in the estimation of this parameter. The ground truth indicates that the translation should be zero, but instead the estimated quantity diverges exponentially to a value of around 10000mm (10m) over the 2.5 second duration of the image sequence. The apparent lack of robustness in the estimation of this parameter may be caused by a number of different factors such as IMU drift or errors in feature tracking. It was found that this component could be improved through applying different values to the tuning parameters although this resulted in significant adverse effects on the other parameters, particularly those in which significant motion is present, which may be considered the more important parameters. Nevertheless, further investigation is required into the cause and eradication of this erroneous result. Figure 5.1 (b) shows the results for the Y-component of fully scaled translation. In this case the results are much more positive since, although some divergence is observed, the final result is only in error by around 7mm. In this case, a likely cause is that the results are driven by the noise in the IMU signal. In Figure 5.1 (c), exceptional performance is observed in the estimation of the Z- component of translation, where it can be seen that the results follow the ground truth very closely for the whole duration of the image sequence. This is a significant validation of the proposed algorithm since this parameter can be considered to be one of the most important. Motion along the Z-axis is the least sensitive component to distinguish in single camera SFM, therefore a strong performance for this parameter is very significant. It is also strongly related to the depth of the feature points, suggesting that it should be possible to obtain accurate structure results using this algorithm.

Figure 5.1 (d) presents the results for the X-component of total rotation. Here it is observed that signal appears to be mostly influenced by noise, which is to be expected since there is no motion for this parameter. However, this result still represents very good performance since the final result is in error by only $1.3\text{x}10^{-3}$rad (0.07 degrees). In Figure 5.1 (e) and (f) the results are again remarkable as they are seen to follow the ground truth very closely. These parameters are also important given that they are parameters in which there is significant motion, and so it is important that they are estimated accurately.

Figure 5.1 (g) presents the results for the estimation of the inverse focal length, in which it is observed that the estimation of this parameter converges rapidly to the correct

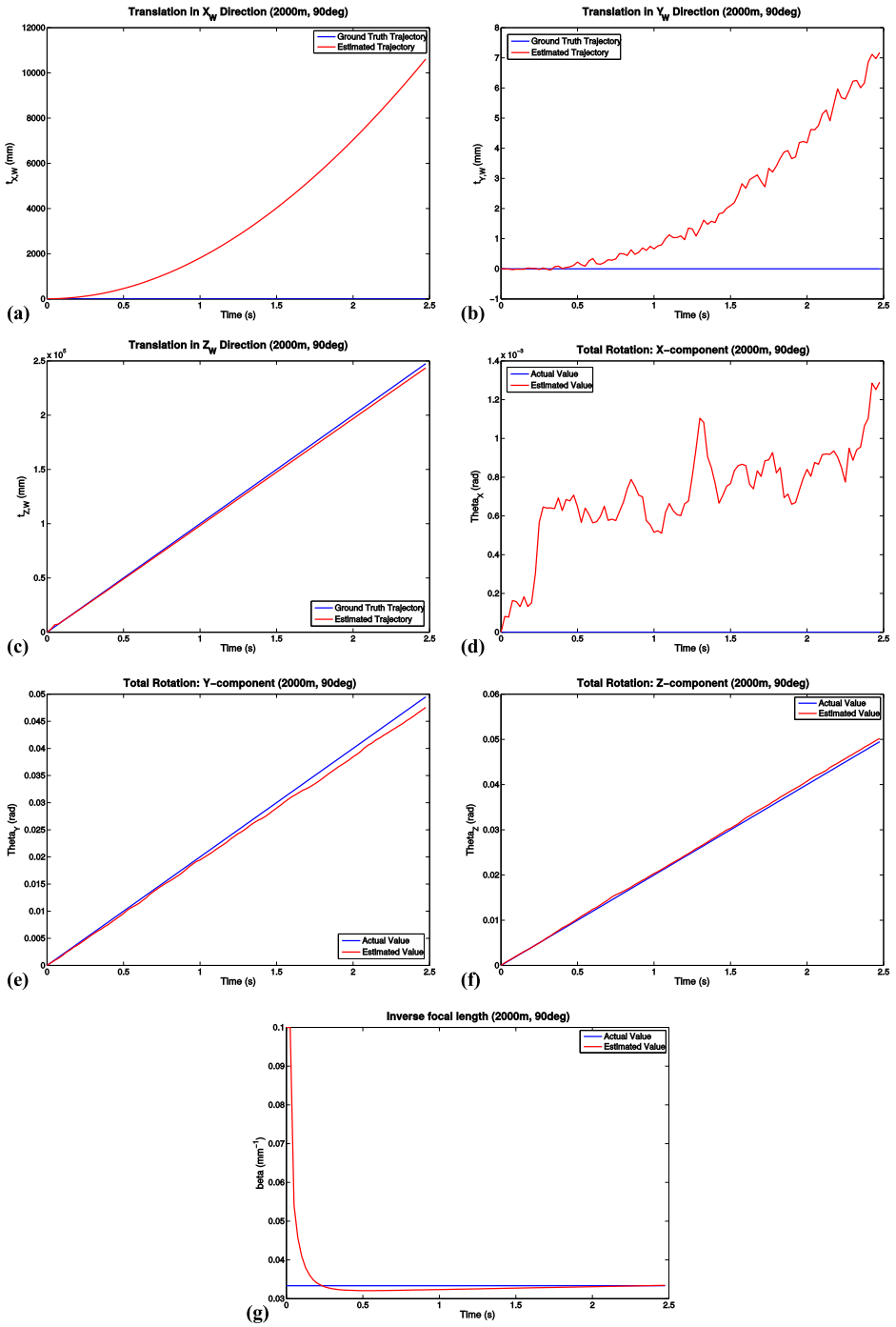


Figure 5.1: EH$_\infty$F motion results from 100 image PANGU sequence (2000m initial altitude, $90°$ trajectory, 100m/s vertical descent velocity, 0.02rad/s rotational velocity about Y_w- and Z_w-axes): a) X_w-component of fully scaled translation; b) Y_w-component of fully scaled translation; c) Z_w-component of fully scaled translation; d) X_w-component of total rotation; e) Y_w-component of total rotation; f) Z_w-component of total rotation; g) Inverse focal length.

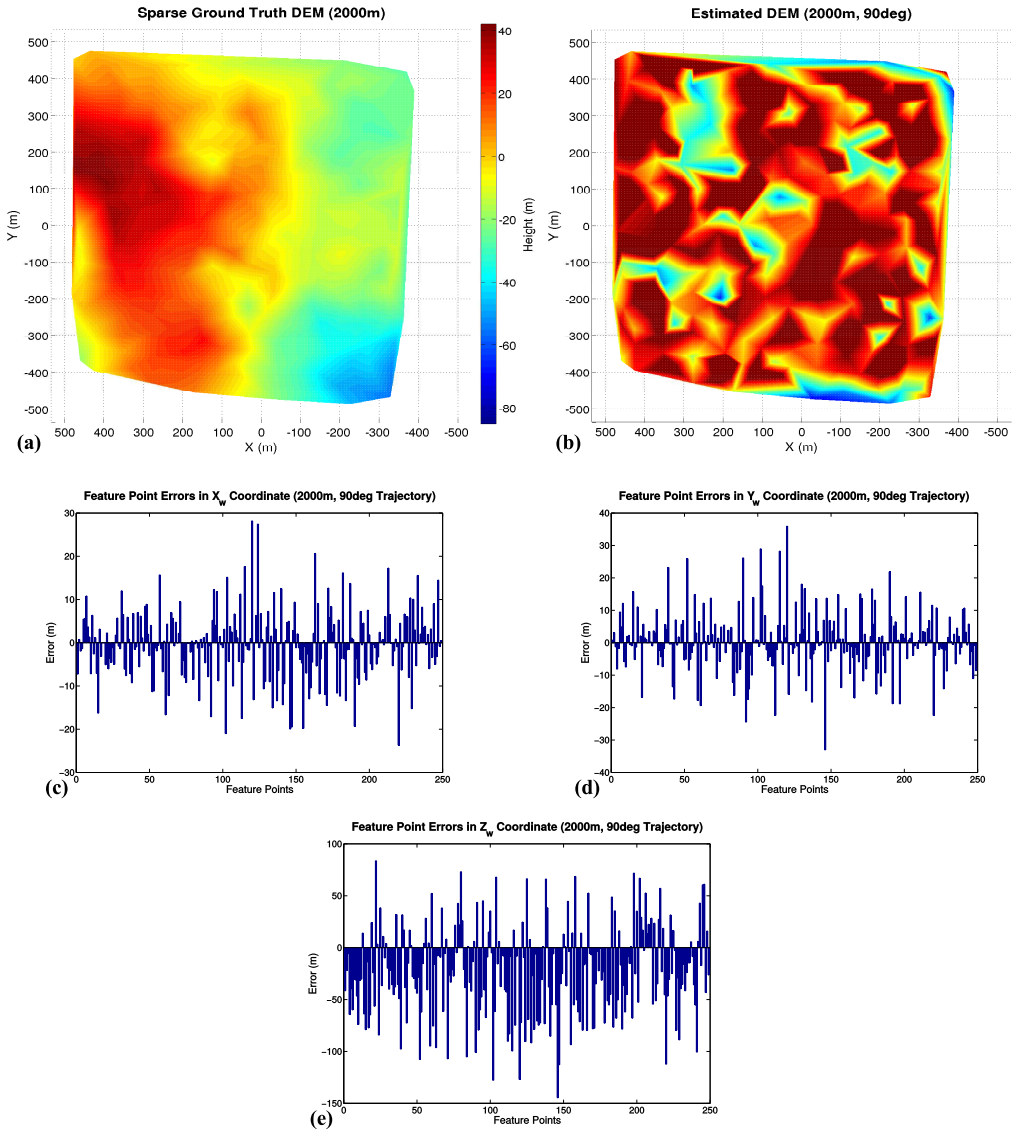


Figure 5.2: EH$_\infty$F structure results from 100 image PANGU sequence (2000m initial altitude, $90°$ trajectory, 100m/s vertical descent velocity, 0.02rad/s rotational velocity about Y_w- and Z_w-axes): a) Sparse ground truth DEM; b) Estimated DEM; c) Errors in estimated X_w feature point 3D position; d) Errors in estimated Y_w feature point 3D position; e) Errors in Z_w feature point 3D position.

value of $1/30$mm. This is also an important result since it is bound together with the Z-component of translation, and it is also what allows metric values of the other components to be obtained without prior calibration of the camera. Therefore, a quick convergence is important for the accuracy of the other motion parameters.

Figure 5.2 (a) presents the ground truth DEM of the terrain that has been artificially reduced in density by selecting those points from a full resolution ground truth DEM that

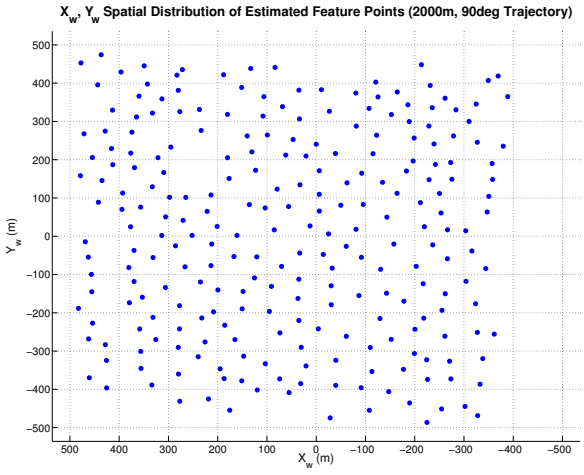


Figure 5.3: World frame $(X_w,\, Y_w)$ positions of IMU-KLT tracked feature points in EH$_\infty$F SFM for 2000m initial altitude

correspond with the tracked feature points. These points can be seen in Figure 5.3. This was done so that a direct comparison can be made with the estimated results, which are shown in Figure 5.2 (b). As can be seen very clearly from the figure, there is a problem with the estimated structure results since it in no way resembles that of the ground truth. This problem is further indicated in Figure 5.2 (e), which shows the errors in the estimated depth for each of the tracked feature points and from this it can be seen that the majority of the errors lie within the range [0m, -100m], with some even approaching -150m. These errors also lead to errors in the $(X_w,\, Y_w)$ coordinates that can be observed in Figure 5.2 (c) and (d), as these are calculated using the camera model equation and the estimated depths. The reasons for this problem are not fully understood especially given that the motion estimation results appear to be very accurate. It may simply be that 100 images did not provide sufficient time for the structure estimates to converge to sufficient accuracy now that there are many more feature points, or it may be an unknown bug in the code. Further investigation is needed to determine what this problem is.

A comparison is made in Figure 5.7 between the motion results of the EH$_\infty$F and some results using the EKF-based SFM algorithm presented in Chapter 4, except that in order to make a direct comparison between the EH$_\infty$F algorithm and the EKF algorithm for the 2000m initial altitude case, the EKF algorithm was run on the same image dataset, which unlike the results in Chapter 4 now also has a rotational velocity of 0.02rad/s about the Y-axis. The EKF is now also run using the IMU-KLT tracked features. However, we first present the results from the EKF in Figure 5.4 before examining the comparison in Figure 5.7.

Figure 5.4 (a) shows the estimation of fully scaled translation in the X-direction. In reality there is no motion in this parameter since the spacecraft is moving solely in vertical direction down towards the surface, however the estimated results show a clear exponential divergence, which by the end of the 100 image sequence has reached a value of above 14000mm (14m). This behaviour was also noted in the results of

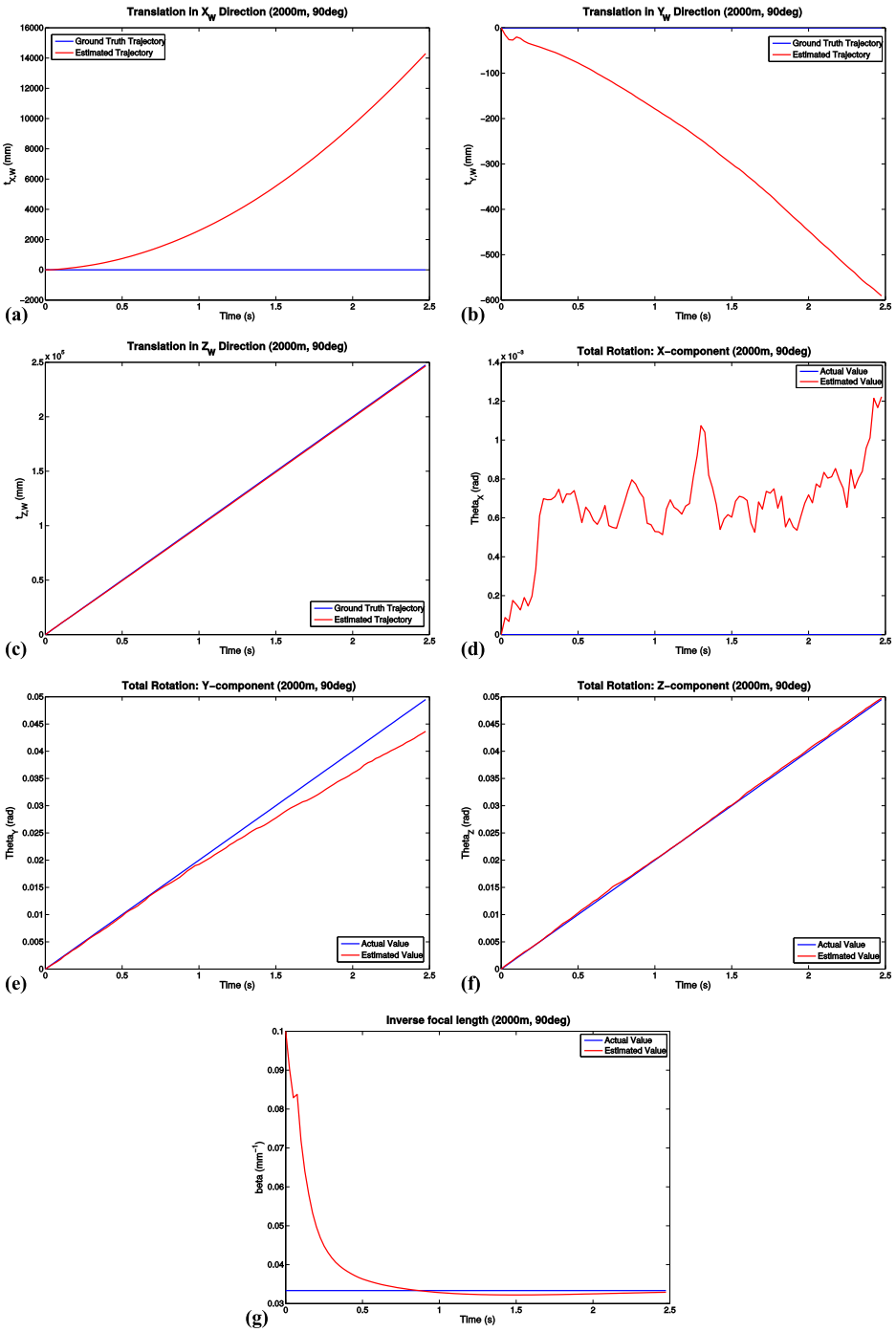


Figure 5.4: EKF-SFM motion results from 2000m initial altitude, $90°$ trajectory, 100m/s vertical descent velocity, 0.02rad/s rotational velocity about Y_w- and Z_w-axes, using the IMU-KLT feature tracker: a) X_w-component of fully scaled translation; b) Y_w-component of fully scaled translation; c) Z_w-component of fully scaled translation; d) X_w-component of total rotation; e) Y_w-component of total rotation; f) Z_w-component of total rotation; g) Inverse focal length.

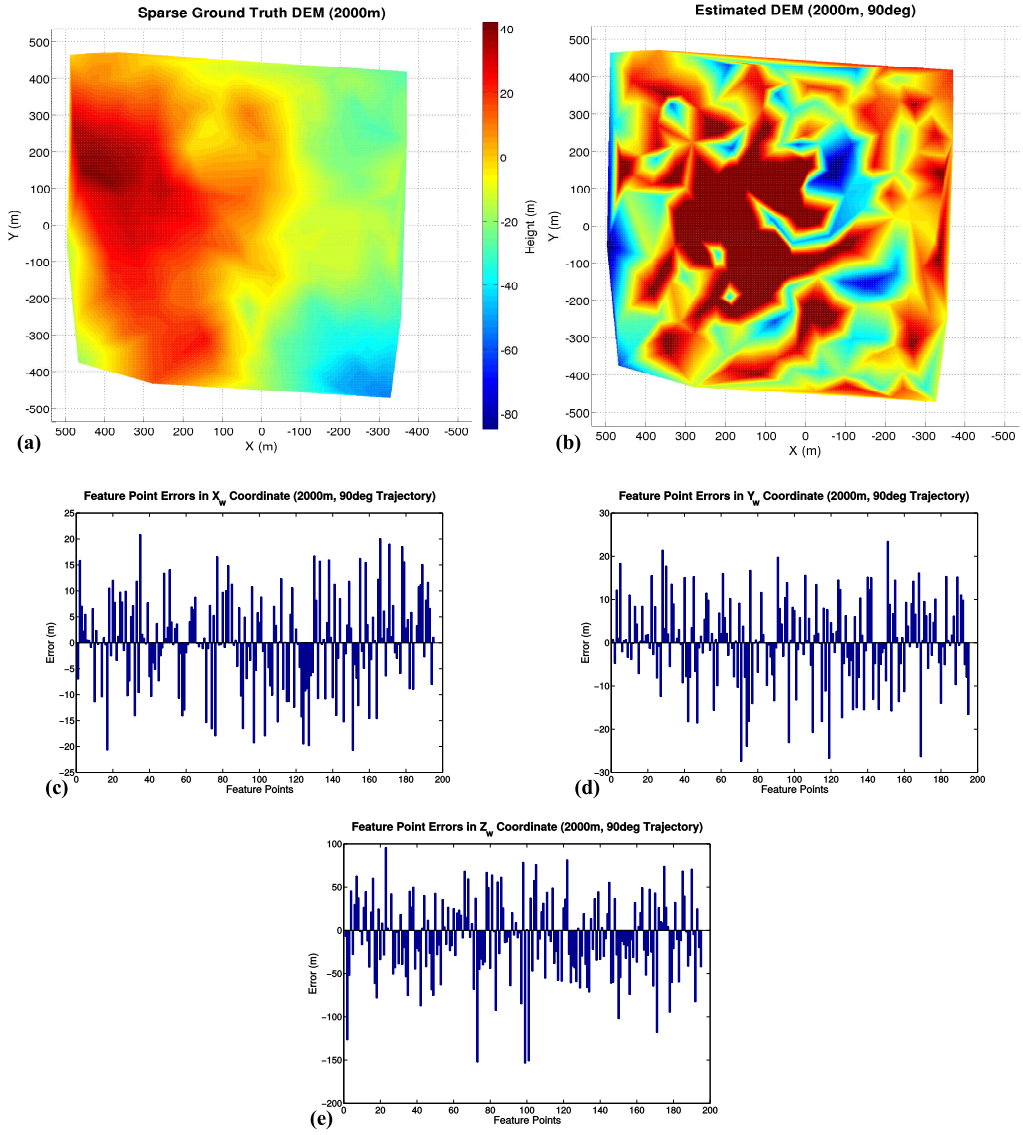


Figure 5.5: EKF-SFM structure results from 2000m initial altitude, $90°$ trajectory, 100m/s vertical descent velocity, 0.02rad/s rotational velocity about Y_w- and Z_w-axes, using the IMU-KLT feature tracker: a) Sparse ground truth DEM; b) Estimated DEM; c) Errors in estimated X_w feature point 3D position; d) Errors in estimated Y_w feature point 3D position; e) Errors in estimated Z_w feature point 3D position.

Chapter 4, Figure 4.9, however now it is slightly more severe. This may be a consequence of now having many more features, which may mean that a number of them are of poor quality, i.e. perhaps containing sufficient tracking error accumulation that over some subset of the features contains a consistent bias, whereas before a large number of features were lost during the tracking and so those few that remained should have been of very high quality and thus have a low accumulation of tracking error in order to have survived that long. Alternatively it could simply be a result of suboptimal tuning as a consequence of the manual tuning procedure that was employed, which may not necessarily result in an optimal tuning. Figure 5.4 (b) also shows divergence for the results of the fully scaled Y-component of translation but not as severe as the X-component. By the end of the sequence the Y-component has diverged to a value of -600mm (-6cm), which is by no means a significant cause for concern, but it is not as good as the previous results in Figure 4.9, which showed a reasonably stable noisy oscillation that was biased around the -1mm line. The reasons for this observed divergence are likely to be the same as those given for the X-component. Figure 5.4 (c) shows the results for the estimation of the Z-component of fully-scaled translation, which demonstrates extremely accurate performance, which is much the same as before. Again this is the most significant parameter since it is the Z-component of translation that has the most significant motion and it is also motion that is difficult to estimate using a single camera because the inability to determine depth from a single image means that there is reduced sensitivity to motion in this direction.

Figure 5.4 (d) shows the results for the estimation of total rotation about the X-axis throughout the image sequence, which shows that the results are very accurate. Even though they are slightly biased, the total rotation oscillates approximately around $7\text{x}10^{-4}$rad, which is an angle of approximately 0.04 degrees when the true rotation is constant at 0 degrees. Therefore this can be considered to be very accurate. Figure 5.4 (e) presents the results for the total rotation about the Y-axis, which, as before, has a constant rotational velocity of 0.02rad/s. It can be seen that the estimated result tracks very closely with the ground truth value for the first 40 images, but then begins to slowly diverge and by the end of the 100 image sequence it deviates by approximately $6\text{x}10^{-3}$rad, which is an angle of only 0.3 degrees. This is again a very strong result, indicating that high accuracy can be achieved, but the slow divergence indicates that further investigation is required to see if this would continue or at a later point would re-converge. It is worth noting however, that this type of rotation is difficult to estimate since it produces image motion similar to vertical translation downwards in the left half of the X-Y plane and upwards in the right half of the X-Y plane, thus there is a reduction in sensitivity to this kind of motion, as mentioned above. The magnitude of rotation is also very small, which means there is potentially a weak signal to noise ratio. For these reasons this result for rotation about the Y-axis is very encouraging. Figure 5.4 (f) shows extremely strong performance in the estimation of the total rotation about the Z-axis, which can be seen to almost perfectly follow the ground truth values. It would be expected that this motion parameter is estimated to a higher level of accuracy than that which was observed for the Y-axis rotation since rotation about the Z-axis leads to much more significant inter-frame image motion, and so therefore this result increases confidence in the tuning parameters arrived at via the manual tuning

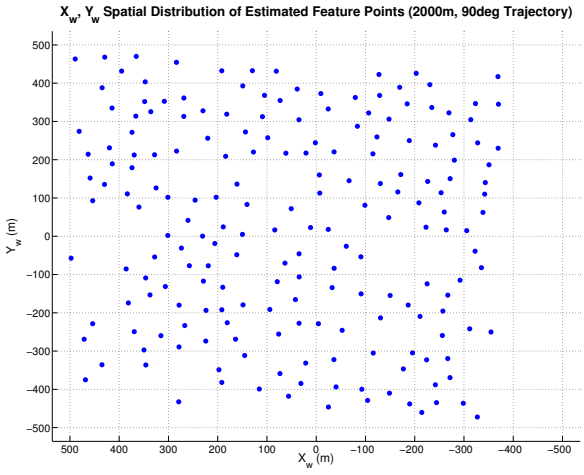


Figure 5.6: World frame (X_w, Y_w) positions of IMU-KLT tracked feature points in EKF SFM for 2000m initial altitude

procedure.

Figure 5.4 (g) presents the results for the estimation of the inverse focal length, which shows that a reasonably rapid and accurate convergence is achieved. This is an important result since the inverse focal length is an important parameter due to its relationship with the Z-component of translation. The inverse focal length is a small number and therefore its combination as a product with t_Z, which is a relatively large number, means that slight errors in B can lead to large errors in t_Z when the two values are separated via division. Therefore the accurate estimation of this parameter also increases confidence that a somewhere close to optimal set of tuning parameters has been found.

Figure 5.5 (a) shows the sparse ground truth DEM from a top-down perspective, which has been derived from a full-resolution (with $1{:}1$ pixel correspondence with the first image in the sequence) ground truth DEM by selecting those points that correspond with the location of the tracked features in the first image of the sequence and interpolating between these points. The feature point positions in the first image of the sequence are can be seen in Figure 5.6, which provides a clear indication of the density of the ground truth and estimated DEMs. Figure 5.5 (b) presents the DEM produced from the estimated depths of the tracked features. Even though the filter has been able to accurately estimate many of the motion parameters it is abundantly clear that there is a problem in the estimation of the scene structure and this can be further seen in Figure 5.5 (e), which shows that the majority of the errors in the estimation of the structure parameters lie in the range ± 70m, with some points having an error in excess of -150m. These errors in depth also lead to the observed errors in the range of approximately ± 20m in the X_w and Y_w coordinates of the feature points, which are calculated using the camera model equation and the estimated feature depths, along with the 2D image coordinates of the feature points in the first image of the sequence. This poor performance for the estimated depths is somewhat surprising given the

performance observed in the previous EKF results in Figure 4.10. The most significant difference between these results and the previous results is the use of the IMU-KLT feature tracker and the fact that this resulted in many more features being tracked over the 100 image sequence so it would possibly seem that the IMU-KLT tracker results in higher levels of tracking error accumulation, which means that there is more significant measurement uncertainty. Alternatively it may simply be a case of insufficient time for the structure parameters to converge given that there may now be a greater degree of measurement error and an increased number of parameters to estimate in the filter state vector. Further tests should be conducted to gain a greater understanding of the limitations of this approach with respect to structure estimation.

To enable a full comparison between the EH$_\infty$F and EKF results using the IMU-KLT features, Figure 5.7 presents the estimation errors for each of the motion parameters and root-mean-square errors for the X_w, Y_w, and Z_w coordinates of the tracked feature points. Looking at Figure 5.7 (a) and (b) it can be seen that for the X and Y components of translation, the EH$_\infty$F provides improved performance for the estimation of these two parameters, in the range of around 4m for the X component and about 0.6m for the Y component. While, in both cases for the EKF and the EH$_\infty$F, divergence was observed for both of these parameters, it is clearly much less severe for the EH$_\infty$F than for the EKF, especially in the case of the Y parameter, therefore it seems that the EH$_\infty$F is keeping the estimation errors more bounded than for the EKF, which is what would be expected given that the EH$_\infty$F is designed to do this. This gives an indication that the EH$_\infty$F is behaving somewhat in line with expectation, even though it was unable to prevent divergence entirely. Figure 5.7 (c) tells a different story, however. Here it can be seen that the EKF gives much better performance and appears to show that the results have satisfactorily converged to a constant error of around 1m, whereas it seems that the EH$_\infty$F results are actually diverging, resulting in a 4m error by the end of the sequence, so in this case the estimation error appears to be unbounded. Its possible that this may converge to a constant error at a later time, and so more investigation is needed to examine this, however it is still clear that the performance of the EH$_\infty$F is unsatisfactory relative to the EKF, even though from Figure 5.1 (c) the results do in fact appear to be very good, and a 4m error at 2000m altitude is more than satisfactory, but only if it does not continue to grow too severely over time, which, unfortunately, Figure 5.7 (c) seems to suggest that it might. This may simply be a sign that the tuning of the EH$_\infty$F is too sub-optimum and that more effort is needed in manually tuning the filter to obtain better results, or again it could be due to the presence of larger than desired tracking error accumulation in the IMU-KLT features, which might suggest that a more robust feature tracking method could be required.

Figure 5.7 (d) presents the error comparison between the EH$_\infty$F and the EKF for the X-component of total rotation. Here it can be seen that the two filters perform very similarly and both result in errors of only a fraction of a degree away from the ground truth value of 0 degrees. In places, however, it can be seen that the EKF gives marginally better performance, however the difference is largely insignificant, and more testing should be performed to be able to draw a stronger conclusion about the relative performance on this motion parameter. For the Y component of total rotation, shown in Figure 5.7 (e), the difference in accuracy is more significant, with the EH$_\infty$F performing better. The EKF

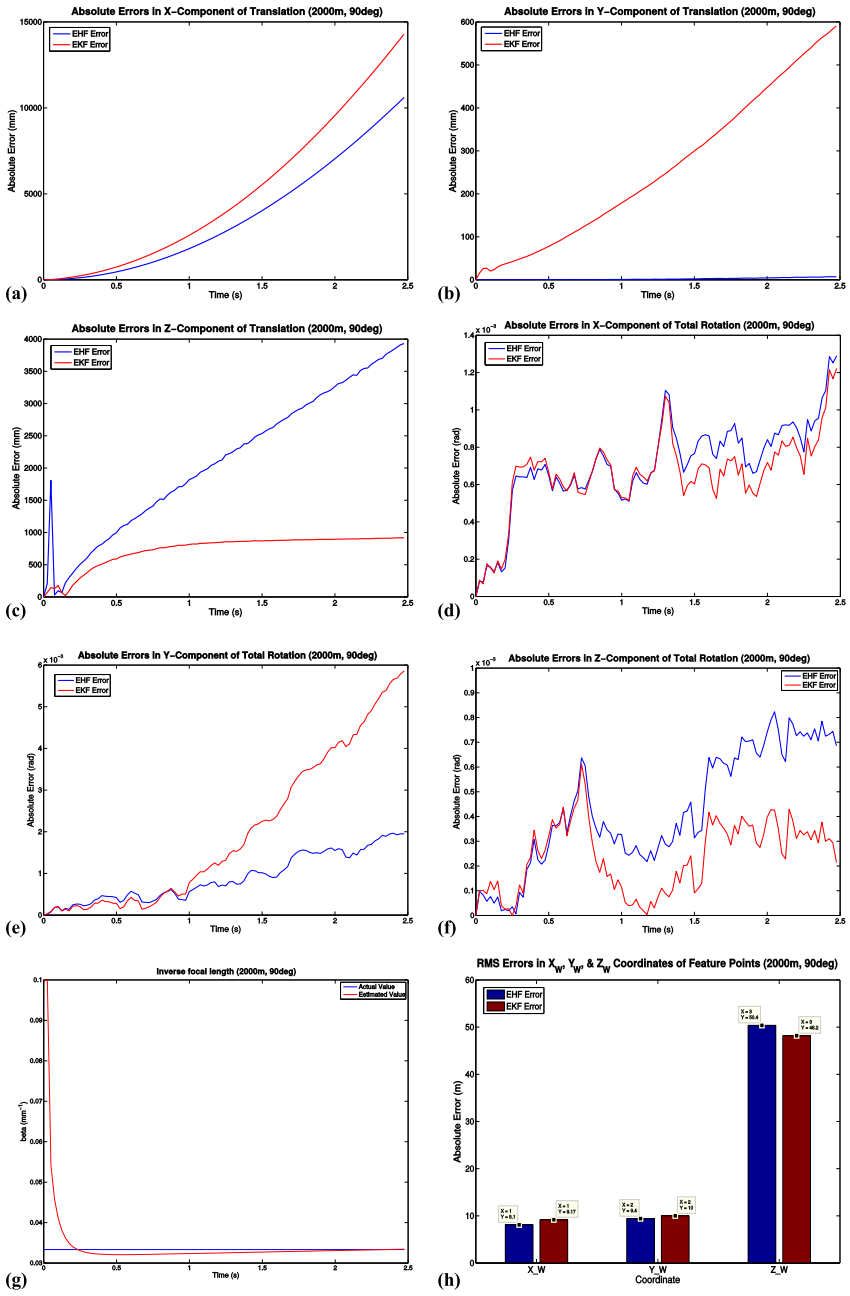


Figure 5.7: Comparison of estimation errors for EH$_\infty$F and EKF based SFM algorithms with IMU-KLT for a 100 image PANGU sequence with initial altitude of 2000m, vertical descent trajectory at a constant velocity of 100m/s, and with 0.02rad/s constant rotational velocity about the Y_w and Z_w axes: (a) Error comparison for X component of translation; (b) Error comparison for Y component of translation; (c) Error comparison for Z component of translation; (d) Error comparison for X component of total rotation; (e) Error comparison for Y component of total rotation; (f) Error comparison for Z component of total rotation; (g) Error comparison for inverse focal length; (h) RMS error comparison for X_w, Y_w, and Z_w 3D scene coordinates of the tracked feature points

results show a clear sign of sudden divergence after the 1s point, whereas the EH$_\infty$F results only increases in error through out the sequence, which means that the EH$_\infty$F is more robust to divergence, as would be expected from the properties of the EH$_\infty$F. Figure 5.7 (f) presents the results for the Z-component of total rotation and shows that both filters perform well for this parameter, as was observed in Figures 5.1 and 5.4. Here, though, the EKF performs slightly better, but on the whole the difference is likely not particularly significant, given that the observed difference is approximately an order of magnitude less than those of Figure 5.7 (e). The reduced accuracy of the EH$_\infty$F compared to the EKF in this case may be an indication that the tuning is not quite optimal and therefore greater accuracy may be achieved by an increased effort in the manual tuning of the filter, however, more investigation is needed to firmly establish the limitations of the new filtering method for this motion parameter.

Figure 5.7 (g) presence the comparison between the two filtering methods for the estimation of the inverse focal length. In this plot it can be seen that both approaches converge to high accuracy very quickly, which, as mentioned before, is important for the estimation of the t_Z component and the structure parameters. The EH$_\infty$F performs better than the EKF for this parameter since it converges more rapidly and achieves a slightly more accurate final estimate than with the EKF.

Figure 5.7 (h) shows the overall rms error for all of the structure parameters for each filtering method. Here it can be seen that both filters produce unsatisfactory estimates of depth as was seen earlier in the DEMs. It actually seems that the EKF gives slightly better performance for the structure (Z_w – which is the most important here since it is the actual parameter that is estimated), however, this is practically irrelevant since they are both very poor, with rms errors of approximately 50m. These results suggest that there is likely something wrong with the tuning parameters or that neither filter is robust enough to estimate this weakly observable parameter, or alternatively (perhaps even additionally) the IMU-KLT algorithm is not accurate enough to provide the quality of measurements required to accurately recover the scene structure.

5.4 Conclusions from H_∞ Filtering

The H_∞ filtering method presented above has demonstrated that it is capable of providing accurate estimates (particularly when the signal to noise ratio is high) for the motion parameters of a spacecraft during descent towards a planetary surface in a short, proof-of-concept test based on a 100 image sequence representing a subset of the descent images that would be produced during a full descent. The EH$_\infty$F is designed from the bottom-up to be more robust than other filtering techniques, such as the ubiquitous EKF. The results obtained using this filtering method do appear to give support to this claim for some of the motion parameters, however, in some of the motion parameters the EH$_\infty$F did not provide a fully clear advantage over the EKF and in others the EKF was shown to produce more accurate estimates. So, while there is some evidence that a more robust and accurate estimation of the motion state of the spacecraft can be achieved using the EH$_\infty$F when compared to the EKF (mostly in reducing or preventing divergence in those parameters that did not possess any actual

motion), the results are somewhat inconclusive. A more thorough investigation into the quality of the tuning parameters for the filter would be required to fully appreciate the limitations of this filtering method, however, given that the EKF is a more straightforward filtering method and is easier to tune, further investigative work into the $EH_\infty F$ was decided not to be a particularly suitable approach to providing a more effective means of improving on the accuracy and robustness of the results already obtained from using the EKF. For this reason, and because neither filtering approach provided anywhere close to satisfactory results for the estimation of the structure parameters, a more sophisticated approach to state estimation was investigated for use in this application, that not only is based on a more robust filtering formulation, but also attempts to adaptively adjust the tuning parameters in the event that they are determined to be sub-optimal, the details of this are presented in the remainder of this chapter.

The $EH_\infty F$ (and EKF) was also used in conjunction with a supposedly more robust method of tracking features throughout the image sequence, known as the IMU-KLT algorithm. While this new feature tracker did appear to be more stable in that it was able to keep many more features alive throughout the 100 image sequence than that of the conventional KLT algorithm, it did not result in an improvement in the estimation of the scene structure. In fact, it resulted in far worse estimates than was observed with conventional KLT and the EKF. Since the EKF was also re-applied, in this chapter, using measurements from the IMU-KLT tracker, and resulted in a severe degradation of the structure estimation, it is possible that the IMU-KLT algorithm may result in a greater quantity of error accumulation in the feature point positions, to the point where it becomes unusable as a feature tracker for 3D reconstruction. While a much more detailed analysis would be required to firmly determine this possibility, it was decided that a potentially more accurate means of tracking features would be beneficial in future work on this problem, and so development work was carried out on implementing a reportedly more robust KLT feature tracking algorithm in order to not only track more features for a longer period of time compared to the conventional KLT algorithm (as was seen with IMU-KLT) but do so in a way that is more immune to error accumulation. The formulation of this alternative KLT algorithm was presented in Chapter 3.

5.5 Adaptive Filtering

The standard Kalman filter [95] was originally developed for the purpose of trajectory estimation for the Apollo program in the 1960s, thus it was developed to be well suited for situations that may contain a number of significant transient modes, such as during the launch of a space vehicle [96]. It would therefore appear that the Kalman filter is capable of performing well in situations in which the process and measurement noise statistics are time-variant, and indeed, the standard formulation allows for this by enabling the noise variances to be expressed as functions of time [96]. However, optimal performance in such a situation is only assured if precise knowledge of the noise characteristics, and how they evolve over time, is known a priori. Exact knowledge of the process noise covariance matrix, $\boldsymbol{Q}$, and the measurement noise covariance matrix, $\boldsymbol{R}$, is of fundamental importance in Kalman filtering as it is these quantities that determine

the relative weighting that should be assigned to the previous knowledge of the state and the new measurements in each time step (i.e. if $\boldsymbol{Q}$ is large compared to $\boldsymbol{R}$, then more confidence will be placed in the new measurements of the system than the propagated previous knowledge of the state, and vice versa). If $\boldsymbol{Q}$ and $\boldsymbol{R}$ are precisely known in every time step, then the previous knowledge of the state and the measurement data can be combined optimally to give a state estimate that is more accurate than through the use of each source of information about the state separately. However, in the majority of real-life situations this knowledge is often very difficult or even impossible to obtain [97], causing the noise covariance matrices to be relegated to tuning parameters that in many cases are assumed to be time-invariant [98] and that must be found through an arduous manual process of trial-and-error. Such a tuning process provides no guarantee that optimal tuning parameters will be found [99], and the time invariant assumption may not be suitably valid in certain situations, leading to sub-optimal performance or potential divergence. The EH$_\infty$F, UKF, and SR-UKF go a long way towards minimising the effects of errors and instability that may result as a consequence of the linearisation step of the EKF or the possibility of non-Gaussian noise, but, as with the KF and EKF, they still contain this fundamental assumption that the process and measurement noise statistics are known in advance for all time steps. Therefore, even with these more sophisticated filtering methods, the $\boldsymbol{Q}$ and $\boldsymbol{R}$ matrices are still often treated as time invariant manual tuning parameters. In our case it is unlikely that the noise statistics can be assumed to be constant since part of the descent trajectory may occur under a parachute and as such may be susceptible to periods of turbulence interspersed within periods relatively smooth motion, and another part of the trajectory may involve a rocket powered descent that is subject to large vibrations and sudden disturbing impulses. Therefore, a suitable method of circumventing this difficulty is likely to be required.

The requirement of precisely known Gaussian measurement and process noise has been a widely known problem in the application of Kalman filtering techniques since the very beginning. Consequently, a large number of methods have been proposed tackle this problem, most of which have made use of the realisation that the measurements used in the correction step of the Kalman filter will not only contain information about the state of the system but also information on the noise statistics [96]. These types of approaches are known as adaptive filtering techniques, which attempt to estimate the unknown noise statistics simultaneously with estimation of the state.

5.5.1 *Early Approaches*

A survey paper written by Mehra [100] in 1972, summarised a number of techniques relating to on-line estimation of the unknown process and measurement noise covariance matrices. At this point the different approaches could be classified into four categories: Bayesian, Maximum Likelihood, Correlation, and Covariance Matching [100]. To discuss these approaches to adaptive filtering it is helpful to layout the problem that they are attempting to solve in more formal terms.

Consider a discrete-time, linear, dynamic system of the form

$$\mathbf{x}_k = \boldsymbol{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{5.8}$$
$$\mathbf{y}_k = \boldsymbol{H}_k\mathbf{x}_k + \mathbf{v}_k \tag{5.9}$$

where $\mathbf{x}_k$ is the $n \times 1$ state vector, $\boldsymbol{F}_k$ is the $n \times n$ state transition matrix, $\mathbf{w}_k$ is a $q \times 1$ process noise vector, $\mathbf{v}_k$ is a $r \times 1$ measurement noise vector, and $\boldsymbol{H}_k$ is the $r \times n$ measurement model matrix. The process and measurement noise vectors, $\mathbf{w}_k$ and $\mathbf{v}_k$ are assumed to be uncorrelated, zero-mean, Gaussian white-noise sequences, thus

$$\begin{aligned} \mathsf{E}[\mathbf{w}_k] &= 0 \\ \mathsf{E}[\mathbf{v}_k] &= 0 \end{aligned}$$

$$\begin{aligned} \mathsf{E}[\mathbf{w}_{k,i}, \mathbf{w}_{k,j}^{\mathsf{T}}] &= \boldsymbol{Q}_k\delta_{ij} \\ \mathsf{E}[\mathbf{v}_{k,i}, \mathbf{v}_{k,j}^{\mathsf{T}}] &= \boldsymbol{R}_k\delta_{ij} \\ \mathsf{E}[\mathbf{w}_{k,i}, \mathbf{v}_{k,j}^{\mathsf{T}}] &= 0 \qquad \forall \quad i,j \end{aligned}$$

where $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are non-negative definite covariance matrices for the process and measurement noise, respectively, and δ_{ij} is the Kronecker delta function ($\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ if $i \neq j$). Now, if $\hat{\mathbf{x}}_{k|l}$ represents an estimate of $\mathbf{x}_k$ given the set of observations $Y^l = \{y_1, \cdots, y_l\}$, and

$$\boldsymbol{P}_{k|l} = \mathsf{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|l})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|l})^{\mathsf{T}}]$$

is the state error covariance matrix at time step k, given the set of measurements up to and including time step l, then the optimal solution to this problem, when $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are exactly known, is given by the standard linear Kalman filter [100]:

Time Update:

$$\hat{\mathbf{x}}_{k|k-1} = \boldsymbol{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} \tag{5.10}$$
$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_{k-1}^{\mathsf{T}} + \boldsymbol{Q}_{k-1} \tag{5.11}$$

Measurement Update:

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathsf{T}}(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k)^{-1} \tag{5.12}$$
$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \boldsymbol{K}_k(\mathbf{y}_k - \boldsymbol{H}_k\hat{\mathbf{x}}_{k|k-1}) \tag{5.13}$$
$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1} \tag{5.14}$$

where $\boldsymbol{K}_k$ is the Kalman gain, which determines how much importance is placed in the measurements vs. the predicted state. When $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are inexactly known, the Kalman filter is sub-optimal and performance degradation occurs, potentially leading to

divergence. If $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are entirely unknown, which is common in practice, then the Kalman filter can no longer be applied. The goal of adaptive Kalman filtering is therefore to obtain simultaneous on-line estimates of $\mathbf{x}_k$ and the unknown noise covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, such that the Kalman filter operates as close to optimality as possible, by adapting the filter to the observation data.

In the Bayesian methods, the approach is to obtain recursive equations for the *a posteriori* joint probability density of $\mathbf{x}_k$ and a parameter vector, $\boldsymbol{\alpha}$, that represents the unknown parameters [100]. Using the multiplication rule of joint probabilities, this *a posteriori* density can be expressed as

$$p(\mathbf{x}_k, \boldsymbol{\alpha}|Y^k) = p(\mathbf{x}_k|\boldsymbol{\alpha}, Y^k)p(\boldsymbol{\alpha}|Y^k) \tag{5.15}$$

where $p(\mathbf{x}_k, \boldsymbol{\alpha}|Y^k)$ is Gaussian with mean $\hat{\mathbf{x}}_{k|k}(\boldsymbol{\alpha})$ and covariance $\boldsymbol{P}_{k|k}(\boldsymbol{\alpha})$, which are given by the Kalman filter equations for a particular $\boldsymbol{\alpha}$ [100]. Note that $p(\boldsymbol{\alpha}|Y^k)$ is the conditional probability density function describing the probability that a particular set of parameters, $\boldsymbol{\alpha}$, is true, given the measurements up to and including the current time step k. An equation for $p(\boldsymbol{\alpha}|Y^k)$ can be obtained by first expressing it as

$$p(\boldsymbol{\alpha}|Y^k) = p(\boldsymbol{\alpha}|\mathbf{y}_k, Y^{k-1}) \tag{5.16}$$

and then applying Bayes theorem to give

$$p(\boldsymbol{\alpha}|\mathbf{y}_k, Y^{k-1}) = \frac{p(\mathbf{y}_k|\boldsymbol{\alpha}, Y^{k-1})p(\boldsymbol{\alpha}|Y^{k-1})}{p(\mathbf{y}_k|Y^{k-1})} \tag{5.17}$$

where the denominator, $p(\mathbf{y}_k|Y^{k-1})$ can be viewed as the marginal conditional probability density function of $\mathbf{y}_k$ obtained from the joint density function by marginalising over all possible $\boldsymbol{\alpha}$'s, i.e.

$$p(\boldsymbol{\alpha}|\mathbf{y}_k, Y^{k-1}) = \frac{p(\mathbf{y}_k|\boldsymbol{\alpha}, Y^{k-1})p(\boldsymbol{\alpha}|Y^{k-1})}{\int_A p(\mathbf{y}_k|\boldsymbol{\alpha}, Y^{k-1})p(\boldsymbol{\alpha}|Y^{k-1})d\boldsymbol{\alpha}}, \tag{5.18}$$

where A is the set of all possible $\boldsymbol{\alpha}$'s [100]. Notice that $p(\mathbf{y}_k|\boldsymbol{\alpha}, Y^{k-1})$ is Gaussian and represents the *a priori* conditional probability density function describing the probability of obtaining the measurement $\mathbf{y}_k$ at time step k given the measurements up to but *not including* time step k, with a particular set of parameters $\boldsymbol{\alpha}$. Thus, it is the conditional probability density function with mean $\boldsymbol{H}_k\hat{\mathbf{x}}_{k|k-1}(\boldsymbol{\alpha})$ and covariance $(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}(\boldsymbol{\alpha})\boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k)$ [100], which are, respectively, the predicted measurement and predicted measurement innovation covariance given by the Kalman filter equations for a particular $\boldsymbol{\alpha}$.

In the conventional Kalman filter, or any other estimation problem where the statistics of the random processes are known, the optimal estimate is given by the conditional

mean, i.e.

$$\hat{\mathbf{x}}_{k|k} = \int_X \mathbf{x}_k p(\mathbf{x}_k|Y^k)\, d\mathbf{x}_k \tag{5.19}$$

where X is the space of all possible $\mathbf{x}_k$, and $p(\mathbf{x}_k|Y^k)$ is the conditional probability density function of $\mathbf{x}_k$ given the data up to and including time step k. Notice that this is the average over all possible $\mathbf{x}_k$, weighted by the conditional probability density function describing the probability that a particular value of $\mathbf{x}_k$ is true given the data.

In the case that the statistics of the random processes are unknown, we must consider the joint conditional probability density function of $\mathbf{x}_k$ and $\boldsymbol{\alpha}$ given the data. By recognising that the conditional density of $\mathbf{x}_k$ can be obtained from the joint conditional density of $\mathbf{x}_k$ and $\boldsymbol{\alpha}$ by integration over the set, A, of all possible $\boldsymbol{\alpha}$, Equation (5.19) can be expressed as [101, 102]

$$\hat{\mathbf{x}}_{k|k} = \int_X \mathbf{x}_k \int_A p(\mathbf{x}_k, \boldsymbol{\alpha}|Y^k)\, d\boldsymbol{\alpha}\, d\mathbf{x}_k, \tag{5.20}$$

which, as in Equation (5.15) can be re-written as

$$\hat{\mathbf{x}}_{k|k} = \int_X \mathbf{x}_k \int_A p(\mathbf{x}_k|\boldsymbol{\alpha}, Y^k) p(\boldsymbol{\alpha}|Y^k)\, d\boldsymbol{\alpha}\, d\mathbf{x}_k. \tag{5.21}$$

By recognising that the integral of $\mathbf{x}_k p(\mathbf{x}_k|\boldsymbol{\alpha}, Y^k)$ over all possible values of $\mathbf{x}_k$, given a parameter vector $\boldsymbol{\alpha}$, represents the optimal estimate of $\mathbf{x}_k(\boldsymbol{\alpha})$, i.e.

$$\hat{\mathbf{x}}_k(\boldsymbol{\alpha}) = \int_X \mathbf{x}_k\, p(\mathbf{x}_k|\boldsymbol{\alpha}, Y^k)\, d\mathbf{x}_k, \tag{5.22}$$

the final form of the optimal estimator is given by

$$\hat{\mathbf{x}}_k = \int_A \hat{\mathbf{x}}_k(\boldsymbol{\alpha})\, p(\boldsymbol{\alpha}|Y^k)\, d\boldsymbol{\alpha}. \tag{5.23}$$

Therefore the optimal estimate is formed by taking the complete set of conditional estimates weighted by the conditional probabilities that the corresponding parameter vector is true (calculated using Equation (5.18)), and integrating over all possible parameter vectors [101]. More specifically, if we had an infinite number of Kalman filters each estimating $\hat{\mathbf{x}}_k(\boldsymbol{\alpha})$ for their own unique value of the parameter vector, $\boldsymbol{\alpha}$, from an infinite set of possible parameter vectors, each of which being weighted by the conditional probability of that $\boldsymbol{\alpha}$ being true given the measurements, then the optimal estimate is given by the integral over all possible parameter vectors. Or, in other words, the scheme converges asymptotically to that of the optimum estimating scheme operating with full knowledge of the previously unknown noise statistics [97].

While this approach seems very appealing from a theoretical point of view it should be noted that, due to the integrals over potentially high dimensional states, it is only feasible in cases where these integrals can be evaluated explicitly, which is only the case under a

number of restrictive assumptions, or if $\boldsymbol{\alpha}$ is known to, or can be assumed to come from a finite set of possible values, in which case the integrals can be replaced by summations over this finite set [100]. However, even if $\boldsymbol{\alpha}$ can be assumed to belong to finite set, the calculation of $\hat{\mathbf{x}}_{k|k}$ for every possible value of $\boldsymbol{\alpha}$ requires many individual Kalman filters to be implemented, which may be very computationally demanding if the set is large. It is also necessary to know the values of all the possible $\boldsymbol{\alpha}$'s in advance [101, 102], as well as the *a priori* marginal probability density of $\boldsymbol{\alpha}$ in order to begin the recursive calculation of $p(\boldsymbol{\alpha}|\mathbf{y}_k, Y^{k-1})$ in Equation (5.18) [100]. These limitations make this approach difficult or impossible in all but a few specialised cases.

The maximum-likelihood approaches begin with an expression such as that in Equation (5.15), and from this a log-likelihood expression is derived. The maximum-likelihood estimate of the unknown state and parameters is calculated by computing the partial derivatives of the log-likelihood equation with respect to the unknown variables, setting these equal to zero and solving for the unknowns. The optimal estimate of the state is then calculated from the Kalman filter using the maximum-likelihood solution of the parameter vector, $\hat{\boldsymbol{\alpha}}$, i.e. $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k}(\hat{\boldsymbol{\alpha}})$. Unfortunately, the equation that must be solved to calculate $\hat{\boldsymbol{\alpha}}$ is non-linear in $\boldsymbol{\alpha}$, and therefore it must be solved iteratively, which becomes prohibitively expensive as the dimensions of the parameter vector increase, since the number of equations to be solved in each iteration increases rapidly with the dimension of $\boldsymbol{\alpha}$ [100]. It is therefore necessary to simplify the equations, which can be done by making a number of potentially restrictive assumptions, such as assuming time invariant systems and that the filter has reached a steady state, for example (see [96, 100] for details on these and additional assumptions), which may limit its usefulness in certain situations and ultimately results in a suboptimal solution.

The correlation-based adaptive estimation methods described in [100] attempt to derive a set of equations relating the system parameters to either the autocorrelation function of the system output $\mathbf{y}_k$ or the autocorrelation function of the measurement innovations $\boldsymbol{\nu} = \mathbf{y}_k - \boldsymbol{H}_k\hat{\mathbf{x}}_{k|k-1}$. Both of these methods require that the system is completely controllable and observable, and in general are only applicable to constant coefficient systems. In particular the output-based correlation approach is also only applicable to cases in which the output signal is stationary or the state transition matrix is stable, whereas the innovation-based method can be applied to systems in which the state transition matrix is unstable [100]. It has also been found that the innovation-based method is more efficient than the output-based method due to $\boldsymbol{\nu}_k$ being less correlated than $\mathbf{y}_k$, which can be understood by the fact that for an optimal filter the innovation sequence is a zero-mean, Gaussian white noise sequence, thus even though the adaptive filtering problem most likely starts in a suboptimal state and hence has an initially correlated innovation sequence, it becomes less correlated over time as it approaches optimality, whereas if the output is correlated it will remain correlated at all times. Both of these methods allow for the calculation of the $\boldsymbol{Q}$ (however, a unique solution of $\boldsymbol{Q}$ may not always be possible) and $\boldsymbol{R}$ matrices or even the direct calculation of the Kalman gain matrix $\boldsymbol{K}$, and due to their relative simplicity, especially in comparison to the Bayesian or maximum likelihood methods, they can be used on-line in real-time. Both methods are also capable of producing asymptotically normal, unbiased

and consistent estimates of $\boldsymbol{Q}, \boldsymbol{R},$ and $\boldsymbol{K}$ [100, 103, 104]. It should be noted that the innovation approach in [103] requires initial estimates of $\boldsymbol{Q}$ and $\boldsymbol{R}$, which are then improved if the filter is found to be suboptimal, whereas the output based method does not require these *a priori* estimates [104]. However, it was suggested in [100] that the output-based method could be used at first to obtain the *a priori* estimates, following which the approach could be changed to that of the innovation-based method to take advantage of its increased efficiency.

A typical approach in covariance matching based techniques is to attempt to make the actual measurement innovation covariance consistent with its theoretical covariance. The actual measurement innovation covariance can be approximated by its sample covariance, sampled over the previous N time steps, given by

$$\boldsymbol{C}_k = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{\nu}_i \boldsymbol{\nu}_i^{\mathsf{T}}. \tag{5.24}$$

The theoretical measurement innovation covariance is that computed by the Kalman filter, i.e.

$$\hat{\boldsymbol{C}}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k. \tag{5.25}$$

By setting Equations (5.24) and (5.25) equal to each other and solving for $\boldsymbol{R}_k$, a value for $\boldsymbol{R}_k$ can be calculated that would make the actual and theoretical measurement innovation covariances consistent. Similarly, by substituting Equation (5.11) into Equation (5.25) and solving for $\boldsymbol{Q}_{k-1}$, an value for $\boldsymbol{Q}_{k-1}$ can be obtained that makes the theoretical and actual covariances consistent. However, in this case a unique solution can only be obtained if $\boldsymbol{H}_k$ is of rank less than n. An important point to note in this method is that it is only approximate since if $\boldsymbol{Q}$ and $\boldsymbol{R}$ are unknown, then $\boldsymbol{P}_{k-1|k-1}$ and $\boldsymbol{P}_{k|k-1}$ do not represent the actual state error covariance, and also the calculated actual covariance is only approximate. Therefore the convergence of this method is doubtful [100]. It has also been reported that the covariance matching method often produces biased estimates of the noise covariance matrices [105].

From the discussion above it would seem that the Bayesian and maximum-likelihood approaches are too restrictive and computationally demanding to apply successfully in all but a few specialised cases. Despite the scepticism over the performance of the covariance matching method in [100], a number of significant advances have been made since the publication of Mehra's paper that appear to have come a long way towards overcoming the original limitations of the method, including some interesting applications to non-linear filtering problems. For this reason the following subsection will examine some of these advances in more detail.

5.5.2 *Covariance Matching and Related Methods*

Many of the above mentioned classical techniques require the unknown noise statistics to be restricted to a set of constant values, or are too restrictive to be applied to non-linear

applications. To counter this, Myers and Tapley [98] developed a covariance matching-type technique that allows for the adaptive estimation of the unknown noise statistics in the case that they may be independent, non-stationary Gaussian white noise sequences that may also have a non-zero mean, and that can also be applied in non-linear state estimation. More formally this technique assumes the following noise statistics:

$$\begin{aligned} \mathsf{E}[\mathbf{w}_k] &= \mathbf{q}_k & \mathsf{E}[(\mathbf{w}_{k,i} - \mathbf{q}_{k,i})(\mathbf{w}_{k,j} - \mathbf{q}_{k,j})^\mathsf{T}] &= \boldsymbol{Q}_k \delta_{ij} \\ \mathsf{E}[\mathbf{v}_k] &= \mathbf{r}_k & \mathsf{E}[(\mathbf{v}_{k,i} - \mathbf{r}_{k,i})(\mathbf{v}_{k,j} - \mathbf{r}_{k,j})^\mathsf{T}] &= \boldsymbol{R}_k \delta_{ij} \end{aligned} \tag{5.26}$$

where $\mathbf{q}_k$ and $\mathbf{r}_k$ are the true means and $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are the true covariance matrices of the process and measurement noise sequences, respectively, at time-step k. The adaptive algorithm is initially derived by treating the problem in batch form over a range of N time steps under the assumption that the unknown parameters $\mathbf{q}_k, \mathbf{r}_k, \boldsymbol{Q}_k,$ and $\boldsymbol{R}_k$ are constant over the N steps. By rearranging Equation (5.8) to obtain an expression for $\mathbf{w}_k$ and rearranging Equation (5.9) to give an expression for $\mathbf{v}_k$, the resulting equations can be regarded as providing a sample of the process noise and measurement noise statistics, respectively, i.e. at time t_j a process noise sample $\mathbf{q}_j$ and an observation noise sample $\mathbf{r}_j$ can be calculated, respectively, from

$$\mathbf{q}_j \equiv \mathbf{w}_{j-1} = \hat{\mathbf{x}}_{j|j} - \boldsymbol{F}_{j-1}\hat{\mathbf{x}}_{j-1|j-1} \tag{5.27}$$

$$\mathbf{r}_j \equiv \mathbf{v}_j = \mathbf{y}_j - \boldsymbol{H}_j\hat{\mathbf{x}}_{j|j-1}. \tag{5.28}$$

Using these samples over the N steps of data, unbiased estimates of $\mathbf{q}_k, \mathbf{r}_k, \boldsymbol{Q}_k,$ and $\boldsymbol{R}_k$ can be obtained from the following equations:

$$\hat{\mathbf{q}}_k = \frac{1}{N}\sum_{j=1}^{N} \mathbf{q}_j \tag{5.29}$$

$$\hat{\mathbf{r}}_k = \frac{1}{N}\sum_{j=1}^{N} \mathbf{r}_j \tag{5.30}$$

$$\hat{\boldsymbol{Q}}_k = \frac{1}{N-1}\sum_{j=1}^{N}\left\{(\mathbf{q}_j - \hat{\mathbf{q}}_k)(\mathbf{q}_j - \hat{\mathbf{q}}_k)^\mathsf{T} - \left(\frac{N-1}{N}\right)(\boldsymbol{F}_{j-1}\boldsymbol{P}_{j-1|j-1}\boldsymbol{F}_{j-1} - \boldsymbol{P}_{j|j})\right\} \tag{5.31}$$

$$\hat{\boldsymbol{R}}_k = \frac{1}{N-1}\sum_{j=1}^{N}\left\{(\mathbf{r}_j - \hat{\mathbf{r}}_k)(\mathbf{r}_j - \hat{\mathbf{r}}_k)^\mathsf{T} - \left(\frac{N-1}{N}\right)(\boldsymbol{H}_j\boldsymbol{P}_{j|j-1}\boldsymbol{H}_j-)\right\}. \tag{5.32}$$

These equations were then reformulated into recursive expressions to enable time-varying statistics to be estimated. Each estimate of the unknown parameters at time t_k is based on the last l_r and l_q noise samples

$\mathbf{r}_j$ $(j = k - l_r + 1, \cdots, k)$ and $\mathbf{q}_j$ $(j = k - l_q + 1, \cdots, k)$, respectively. Thus we have the following recursive equations for the estimated mean values, $\hat{\mathbf{r}}_k$ and $\hat{\mathbf{q}}_k$:

$$\hat{\mathbf{r}}_k = \hat{\mathbf{r}}_{k-1} + \frac{1}{l_r}(\mathbf{r}_k - \mathbf{r}_{k-l_r}) \tag{5.33}$$

$$\hat{\mathbf{q}}_k = \hat{\mathbf{q}}_{k-1} + \frac{1}{l_q}(\mathbf{q}_k - \mathbf{q}_{k-l_r}) \tag{5.34}$$

where the term in parenthesis in each of these expressions represents the incorporation of a new noise sample at time step k and the discarding of the oldest sample, each of which must be scaled by dividing by the total number of samples under consideration in accordance with the definition of the sample mean. For the noise covariances we have the following expressions:

$$\begin{aligned}\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{R}}_{k-1} + \frac{1}{l_r - 1}\{&(\mathbf{r}_k - \hat{\mathbf{r}}_k)(\mathbf{r}_k - \hat{\mathbf{r}}_k)^{\mathsf{T}} - (\mathbf{r}_{k-l_r} - \hat{\mathbf{r}}_k)(\mathbf{r}_{k-l_r} - \hat{\mathbf{r}}_k)^{\mathsf{T}} \\ &+ \frac{1}{l_r}(\mathbf{r}_k - \mathbf{r}_{k-l_r})(\mathbf{r}_k - \mathbf{r}_{k-l_r})^{\mathsf{T}} + \left(\frac{l_r - 1}{l_r}\right)[\boldsymbol{\rho}_{k-l_r} - \boldsymbol{\rho}_k]\}\end{aligned} \tag{5.35}$$

$$\begin{aligned}\hat{\boldsymbol{Q}}_k = \hat{\boldsymbol{Q}}_{k-1} + \frac{1}{l_q - 1}\{&(\mathbf{q}_k - \hat{\mathbf{q}}_k)(\mathbf{q}_k - \hat{\mathbf{q}}_k)^{\mathsf{T}} - (\mathbf{q}_{k-l_q} - \hat{\mathbf{q}}_k)(\mathbf{q}_{k-l_q} - \hat{\mathbf{q}}_k)^{\mathsf{T}} \\ &+ \frac{1}{l_q}(\mathbf{q}_k - \mathbf{q}_{k-l_q})(\mathbf{q}_k - \mathbf{q}_{k-l_q})^{\mathsf{T}} + \left(\frac{l_q - 1}{l_q}\right)[\boldsymbol{\Omega}_{k-l_q} - \boldsymbol{\Omega}_k]\}\end{aligned} \tag{5.36}$$

where

$$\boldsymbol{\rho}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} \tag{5.37}$$

$$\boldsymbol{\Omega}_k = \boldsymbol{F}_{k-1} \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_{k-1}^{\mathsf{T}} - \boldsymbol{P}_{k|k}. \tag{5.38}$$

A couple of implementation considerations were highlighted in [98] to improve numerical stability of the algorithm. Firstly, it is possible that the covariance estimators may become negative definite, especially when a small amount of data has been processed, thus the diagonal components of $\hat{\boldsymbol{Q}}_k$ and $\hat{\boldsymbol{R}}_k$ are always reset to the absolute values of the estimates. Secondly, during filter initialisation, the noise samples $\mathbf{r}_j$ and $\mathbf{q}_j$ are poor indicators of the local noise environment. To counter this a fading memory approach is adopted in which successive noise samples are multiplied by a growing weight factor given by

$$\omega_k = (k-1)(k-2)\cdots(k-\beta)/k^{\beta}, \tag{5.39}$$

which has the property $\lim_{k\to\infty} \omega_k = 1$, and the use of invalid noise samples is delayed for the first β time-steps.

The use of a limited memory filter, provided by the moving window within which the noise is sampled, along with the additional fading memory mechanism applied to the noise samples to reduce the impact of the early samples while the filter is in the initial transient stage, has a number of practical advantages. The fading memory mechanism reduces the impact of the, most likely erroneous, *a priori* noise statistics that must be provided during filter initialisation, while the limited memory mechanism can become especially important in non-linear estimation problems. State estimation of non-linear systems often makes use of the extended Kalman filter (EKF), in which the system is linearised about the current state estimate in each time step. This may introduce significant linearisation errors in some cases. Couple this with the possibility of modelling errors, computation errors, etc., then the use of the EKF over long time intervals can result in a situation where the state error covariance matrix becomes unrealistically small and optimistic. Therefore, the filter gain becomes unacceptably small such that subsequent measurements are effectively ignored, thus the model errors may drive the estimate far away from the values predicted by theory. An example of where this often occurs is in the determination of space vehicle trajectories. Moving window filters are capable of rectifying this problem by discarding past data that is beyond the accuracy or predictability range of the model [106].

Kirlin and Moghaddamjoo [107], drawing on inspiration from [108, 109], presented an extension to the method in [98] that offered increased robustness for non-linear systems with non-Gaussian noise by replacing the estimators of mean and covariance with estimators of location and spread based on the median (instead of mean) and the bi-weight measure of [109] (instead of covariance). Moghaddamjoo and Kirlin [110] proposed a further improvement over that of [98] and their previous work [107], by recognising that there is an inherent instability in the formulation of the Myers and Tapley [98] method. This instability arises from the fact that the estimates of the unknown parameters are directly related to the estimates of the state vector, which itself is a function of the unknown parameters. This establishes a positive feedback loop such that any abrupt change in the input forcing function or the noise environment will cause a rapid degradation of the state estimates from their optimum values, which in turn causes a rapid increase in the estimates of the noise covariances, which then causes a further degradation in the state estimates. The existence of this closed loop severely limits the applicability of the algorithm, especially in situations that involve long time runs [110]. The robust version of this algorithm presented in [107] offers some improvement, allowing the algorithm to be run for longer periods of time, but nevertheless still ultimately suffers from the same source of instability. To overcome this limitation, a limited memory curve fitting algorithm was developed in [110] that is robust to measurement error outliers and provides optimum (in the weighted least-square error sense) estimates of the input forcing functions and measurement noise covariances, under the assumption that the mathematical model of the system is correct. This provides an estimation of the unknown parameters that are independent of the state estimates, therefore bypassing the source of instability. They also apply a stochastic approximation method to adjust the process noise covariance matrix in such a way that the statistics of the filter's residuals approaches that of the optimum Kalman filter, thus introducing a stabilizing negative feedback in the statistics of the residual

sequence [110].

By applying the principles of maximum likelihood estimation to the innovation sequence over a moving window, Mohamed and Schwartz [111] derived equations for adaptively estimating the process and measurement noise covariance matrices that are remarkably similar to those used in covariance matching methods. In the case of the measurement noise covariance matrix the expression obtained is identical to that of the classical approach presented in [100]. This equation is repeated here for convenience, along with that obtained for estimating the process noise covariance matrix. The equation for adaptively estimating $\boldsymbol{R}_k$ is

$$\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{C}}_{\nu_k} - \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} \tag{5.40}$$

where, as before

$$\hat{\boldsymbol{C}}_{\nu_k} = \frac{1}{N} \sum_{j=j_0}^{k} \boldsymbol{\nu}_j \boldsymbol{\nu}_j^{\mathsf{T}} \tag{5.41}$$

where $j_0 = k - N + 1$ is the oldest epoch within the estimation window, and k is the time step for which we are estimating the measurement noise covariance. By applying the same method to the filter residual sequence instead of the innovation sequence an alternative equation for $\hat{\boldsymbol{R}}_k$ was also derived, giving

$$\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{C}}_{\nu_k} + \boldsymbol{H}_k \boldsymbol{P}_{k|k} \boldsymbol{H}_k^{\mathsf{T}} \tag{5.42}$$

where $\hat{\boldsymbol{C}}_{\nu_k}$ is again given by Equation (5.41), but in this case

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \boldsymbol{H}_k \hat{\mathbf{x}}_{k|k} \tag{5.43}$$

instead of

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \boldsymbol{H}_k \hat{\mathbf{x}}_{k|k-1} \tag{5.44}$$

as in the previous case. This form of the adaptive estimator for $\boldsymbol{R}$ offers greater numerical stability since it guarantees $\hat{\boldsymbol{R}}_k$ to be at least positive semi-definite, which is a requirement for covariance matrices, whereas the previous equation could result in a negative definite estimate of $\hat{\boldsymbol{R}}_k$. The equation for adapting $\boldsymbol{Q}_k$ is given by

$$\hat{\boldsymbol{Q}}_k = \frac{1}{N} \sum_{j=j_0}^{k} \Delta\mathbf{x}_j \Delta\mathbf{x}_j^{\mathsf{T}} + \boldsymbol{P}_{k|k} - \boldsymbol{F}_{k-1} \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_{k-1}^{\mathsf{T}} \tag{5.45}$$

where $\Delta\mathbf{x}$ is the difference between the predicted state vector and the state vector after the measurement update, i.e.

$$\Delta\mathbf{x}_k = \hat{\mathbf{x}}_{k|k} - \hat{\mathbf{x}}_{k|k-1}. \tag{5.46}$$

Under steady state conditions, i.e. the state error covariance matrix is constant, the second and third terms in Equation (5.45) can be neglected, therefore $\hat{\boldsymbol{Q}}_k$ can be approximated by

$$\hat{\boldsymbol{Q}}_k = \boldsymbol{K}_k \hat{\boldsymbol{C}}_{\nu_k} \boldsymbol{K}_k^{\mathsf{T}}. \tag{5.47}$$

In formulating this method a number of assumptions were made, which are stated in [111] as follows:

1. The filter states $\mathbf{x}$ are independent of the adaptive parameters $\boldsymbol{\alpha}$, i.e. $\partial \mathbf{x} / \partial \boldsymbol{\alpha} = 0$.
2. The state transition matrix $\boldsymbol{F}$ and the measurement model matrix $\boldsymbol{H}$ are time invariant and independent of $\boldsymbol{\alpha}$.
3. The innovation sequence is a white and ergodic sequence within the estimation window.
4. The covariance matrix $\boldsymbol{C}_\nu$ (through $\boldsymbol{\nu}$) is the key to adaptation and hence is the $\boldsymbol{\alpha}$-dependent parameter.

Providing the state vector has not been augmented with some parameter or parameters that depend on the unknown noise statistics, then assumption 2 is likely to be valid in many cases. Assumption 3 may be approximately valid providing the moving window is not too small. However, while the true state values will not be a function of the unknown parameters, providing the state is not augmented with parameters dependent upon $\boldsymbol{\alpha}$, the estimated state vector will be dependent on $\boldsymbol{\alpha}$, by virtue of the Kalman filter equations. Therefore it is likely that this method will suffer from the same instability as that in [98] and [107], as pointed out in [110]. It should also be noted that, as with all the covariance matching methods, the equation for $\hat{\boldsymbol{R}}_k$ was derived by assuming that $\hat{\boldsymbol{Q}}_k$ is already fully known, and vice-versa. Therefore simultaneously estimating both $\boldsymbol{Q}$ and $\boldsymbol{R}$ may not always be feasible [112]. Nevertheless, the method developed by Mohamed and Schwartz has been implemented by a number of authors [113–115] that have reported significant improvements over the conventional Kalman filter.

Ding et al [116] applied the principles of least squares estimation to arrive at the same covariance matching equations derived in [111]. In their work they noted that the estimation converges to the true value as the window size becomes larger, providing the assumption of an ergodic and stationary stochastic process is valid, which is equivalent to some of the assumptions made in [111]. However it was pointed out that a large window size and the fulfilment of the stationary condition can be contradictory requirements to the goals of adaptive state estimation in many scenarios — i.e. it is often because of a non-stationary external environment that we wish to apply adaptive estimation. Therefore, an additional aspect of the work presented in [116] was to focus on the trade off between estimation stability and estimation accuracy when selecting an appropriate window size. For the INS/GPS data fusion strategy used in [116] it was found that a window size covering around 500 epochs gave the closest to optimal performance, and window sizes smaller than around 250 epochs resulted in unstable estimation. The issue of stability due to simultaneous estimation of of both $\boldsymbol{Q}$ and $\boldsymbol{R}$ was also raised due to the estimation of one requiring knowledge of the other. This lead on to some further work, presented in [117, 118], in which a simple process noise covariance scaling algorithm was developed that was said to be more robust to covariance

estimation bias due to fewer parameters being involved in the adaptive tuning. The concept behind this technique is to adaptively adjust the overall magnitude of $\hat{\boldsymbol{Q}}_k$ by calculating a single scalar multiplicative value from the innovation sequence that is then applied directly to the process noise covariance matrix. This adjusts the relative importance of the measurements compared to the state predictions via its influence on the calculation of the filter gain. The results of a comparison between the conventional extended Kalman filter, an EKF with covariance matching, and an EKF with the proposed covariance scaling method were presented in [117, 118], where the authors claimed a significant improvement in the filtering performance, however it must be said that, in this author's opinion, none of the filtering methods appeared to show entirely convincing performance in this application, even though the method is theoretically sound. A somewhat similar covariance scaling method was presented in [114, 115], where in this case a performance increase over the covariance matching technique was observed more convincingly. Further investigations were carried out in [119] into the relative performance of the process noise covariance scaling method proposed in [117, 118], the two covariance matching adaptive filter approaches for estimating the measurement noise covariance matrix (one based on the innovation sequence (Equation (5.40)) and one using the residual sequence (Equation (5.42))), and the conventional Kalman filter. The performance of these methods was analysed by examining the RMS values of the state error covariance matrix $\boldsymbol{P}$ over the course of the estimation process. All the adaptive methods were shown to outperform the conventional Kalman filter, the covariance scaling method appeared to be more stable but did not provide the level of accuracy achieved with the two covariance matching approaches. The two covariance matching methods gave similar performance in terms of the final estimation accuracy once the filter had converged, with one case of the residual-based method outperforming the innovation-based method. However, the residual approach offered much more stability during the initial transient phase and appeared to converge more rapidly than the innovation-based method, therefore it was considered that the residual-based method was the best performing approach. An investigation into the window size was also shown to have minimal impact providing the dynamics environment is not changing too rapidly [119].

The Myers and Tapley method [98] has been revised and successfully applied to extended Kalman filtering by a number of additional authors, e.g. [120] where it was applied in estimating power system frequency deviations and its rate of change in the presence of significant transients and high frequency noise that occurs during sudden power system overloads, [121] in which it was applied to the localisation of a mobile robot using measurements from sonar sensors, but perhaps most significantly for the problem investigated in this current work, in [122] and [123] it was successfully applied in an image feature tracking computer vision algorithm for visual servoing. The approach in [123] included additional modifications to tailor the adaptive algorithm specifically for the visual motion estimation problem by separating the measurement noise statistics into separate components for the two principal directions in the image plane, therefore allowing different noise covariances to be estimated for the x and y locations of the tracked feature points. This slightly reformulated algorithm was tested alongside the conventional EKF in the estimation of position and orientation of imaged objects in the

scene with known true position and orientation to provide a ground truth. It was observed that the adaptive EKF provided slight improvements in the estimation of position and significant improvement in the estimation of orientation compared to the non-adaptive EKF. Li et al [124] also applied the Myers and Tapley method to a non-linear filtering problem, but in this case they applied it to the unscented Kalman filter (UKF) and found that it offered an improvement in the estimation accuracy compared to the non-adaptive UKF, which in turn was better than the standard EKF. It was also stated, but not shown, that the adaptive UKF outperforms the adaptive EKF.

An approach known as the adaptive fading Kalman filter (AFKF) or adaptive 2-stage Kalman filter, that is similar to the covariance scaling method described in [117, 118], was developed by Kim et al in [125] and later analysed for its stability in [126]. Following on from this the same technique was further developed and applied to the extended Kalman filter in [127, 128] to give an approach known as the adaptive fading EKF (AFEKF) or adaptive 2-stage EKF. A stability analysis of this AFEKF was also carried out in [129]. In these approaches a forgetting factor, λ_k, is applied in the calculation of the predicted state error covariance matrix:

$$\boldsymbol{P}_{k|k-1} = \lambda_k \left[\boldsymbol{F}_{k-1} \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_{k-1}^{\mathsf{T}} + \boldsymbol{Q}_{k-1} \right], \tag{5.48}$$

which is similar to the method in [117, 118], except that it is applied to the entire equation for $\boldsymbol{P}_{k|k-1}$ instead of just to $\boldsymbol{Q}_{k-1}$. In addition to this, a scalar parameter, α_k, is computed based on the degree to which the theoretical innovation covariance matches the approximated actual innovation covariance according to the following expression:

$$\alpha_k = \max \left\{ 1, \, trace(\hat{\boldsymbol{C}}_k \boldsymbol{C}_k^{-1}) \right\} \tag{5.49}$$

or

$$\alpha_k = \max \left\{ 1, \, \frac{trace(\hat{\boldsymbol{C}}_k)}{trace(\boldsymbol{C}_k)} \right\}. \tag{5.50}$$

where

$$\hat{\boldsymbol{C}}_k = \frac{1}{N} \sum_{j=j_0}^{k} \boldsymbol{\nu}_j \boldsymbol{\nu}_j^{\mathsf{T}} \tag{5.51}$$

and

$$\boldsymbol{C}_k = \mathsf{E} \left[\boldsymbol{\nu}_k \boldsymbol{\nu}_k^{\mathsf{T}} \right] = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k \tag{5.52}$$

This is then applied, along with λ_k, in the calculation of the Kalman gain:

$$\boldsymbol{K}_k = \frac{\lambda_k}{\alpha_k} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} \left[\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k \right]^{-1}. \tag{5.53}$$

It was stated in [128] that it can either be assumed that λ_k is almost equal to α_k, therefore the state error covariance is scaled by α_k (which is similar to [117, 118]), or λ_k can be

set equal to 1, in which case the state error covariance is unscaled and the Kalman gain is decreased by $1/\alpha_k$. Using this adaptive fading approach significant improvements were observed compared to the conventional KF or EKF and a number of conditions were established for which the AFKF and AFEKF provides stable estimation (see [126] and [129], respectively, for details).

The adaptive fading method developed by Kim et al [125–129] was applied by Fathabadi et al [130] to the EKF and UKF and the relative performance of these two methods were compared with each other and with the conventional, non-adaptive versions of the EKF and UKF, in the non-linear estimation of the relevant parameters of a continuous stirred tank reactor. In all cases it was found that the two adaptive techniques significantly outperformed the two non-adaptive filters, where out of the non-adaptive methods the UKF gave better performance than the EKF, and out of the adaptive methods the AFUKF gave better performance for all the estimated parameters than the AFEKF.

Zhou et al [131] proposed an adaptive unscented Kalman filter (AUKF) based on the residual-sequence form of the adaptive algorithm proposed by Mohamed and Schwarz [111] (Equation (5.42)) and then extended this AUKF by adding a particle filter-based refinement stage to improve the accuracy of the *a posteriori* estimates from the AUKF. The resultant algorithm was named the adaptive unscented particle filter (AUPF). In order to apply the Mohamed and Schwarz method to the UKF, a slight modification of Equation (5.42) (which is repeated here for clarity) was required. Equation (5.42),

$$\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{C}}_{\nu_k} + \boldsymbol{H}_k \boldsymbol{P}_{k|k} \boldsymbol{H}_k^{\mathsf{T}}, \tag{5.54}$$

contains the *a posteriori* information about the state at time step k, via the *a posteriori* state error covariance matrix and the approximated *a posteriori* true innovation covariance matrix given by

$$\hat{\boldsymbol{C}}_{\nu_k} = \frac{1}{N} \sum_{j=j_0}^{k} \boldsymbol{\nu}_j \boldsymbol{\nu}_j^{\mathsf{T}} \tag{5.55}$$

where

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \boldsymbol{H}_k \hat{\mathbf{x}}_{k|k} \tag{5.56}$$

for a linear measurement model, or for a non-linear measurement model $\mathbf{h}_k\left(\mathbf{x}_{k|k}\right)$

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k}\right), \tag{5.57}$$

where in this case the $\boldsymbol{H}_k$ in Equation (5.54) would represent the measurement model Jacobian matrix. The second term on the right-hand-side of Equation (5.54) represents the *a posteriori* measurement innovation covariance, which we shall denote as $\boldsymbol{P}^{+}_{y_k y_k}$. Hence, Equation (5.54) becomes

$$\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{C}}_{\nu_k} + \boldsymbol{P}^{+}_{y_k y_k} \tag{5.58}$$

By examining the UKF equations (see Section 5.6) it can be seen that $\boldsymbol{P}^{+}_{y_k y_k}$ can be calculated by

$$\boldsymbol{P}^{+}_{y_k y_k} = \sum_{i=0}^{2n} W_i^c \left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k,i}\right) - \hat{\mathbf{y}}_{k|k}\right]\left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k,i}\right) - \hat{\mathbf{y}}_{k|k}\right]^{\mathsf{T}} \tag{5.59}$$

where

$$\hat{\mathbf{y}}_{k|k} = \sum_{i=0}^{2n} W_i^m \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k,i}\right), \tag{5.60}$$

and where the $\hat{\mathbf{x}}_{k|k,i}$ represent the sigma points drawn from the UKF a posteriori estimates, i.e. [131]

$$\hat{\mathbf{x}}_{k|k,0} = \hat{\mathbf{x}}_{k|k} \tag{5.61}$$

$$\hat{\mathbf{x}}_{k|k,i} = \hat{\mathbf{x}}_{k|k} - \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k}}\right) \tag{5.62}$$

$$\hat{\mathbf{x}}_{k|k,i+n} = \hat{\mathbf{x}}_{k|k} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k}}\right) \tag{5.63}$$

$$i = 1, \cdots, n.$$

Using this extension of the Mohamed and Schwarz [111] method to the UKF, the performance of the AUKF and AUPF, as well as the AEKF, were compared in an INS/GPS data fusion algorithm in [131]. The results showed that the AUKF outperformed the AEKF and that the AUPF consistently outperformed both.

5.6 Master-Slave Square Root Unscented Kalman Filtering

In this section we present a different type of adaptive filtering algorithm compared to those discussed in Section 5.5. Instead of using specific recursive equations to estimate the unknown statistics from the innovation sequence, we instead elect to use a second filter running in parallel to the main structure from motion filter. This type of filtering arrangement is known as a master-slave configuration, where the master filter is responsible for estimating the structure parameters and the slave filter operates on the innovation sequence of the master filter to recursively estimate the unknown noise statistics. The backbone filtering method used in this master-slave algorithm is that of the square-root unscented Kalman filter, due to the numerical benefits discussed previously. Note that from now on, we are only estimating the structure parameters and are no longer estimating the motion parameters. That is, for the remainder of this thesis we are assuming motion is fully known. This approach was adopted under the advisement of ESA, that stated that it could be reasonable to assume that some other

spacecraft subsystem would be providing estimates of the landing craft's motion during descent. In the absence of knowledge about the uncertainties present in the supplied motion information, we have made a simplifying assumption that the motion is known without error so as to focus solely on developing our algorithms specifically for structure estimation without concern for simultaneous accurate estimation of motion.

5.6.1 Master-Slave SR-UKF Formulation

In the structure from motion algorithm presented in Chapter 4, it was stated that because the structure parameters are stationary in the world frame, since we are observing a rigid scene, there is no process model or process noise for these parameters. Therefore, we have complete knowledge of the process noise statistics, i.e. $\boldsymbol{Q}_k^x = \mathbf{0},\ \forall\, k$. Therefore, in this section we describe an adaptive algorithm that makes use of a second filter running alongside the main SFM filter, that takes the SFM measurement innovation sequence as its measurement signal and recursively estimates the measurement noise covariance matrix $\boldsymbol{R}_k^x$ for the main filter. This type of filtering arrangement is known as a master-slave configuration and was first developed based on the UKF by Song et al [132] and further investigated in [133, 134]. The purpose of this particular algorithm is not only to estimate the unknown noise statistics but also to allow for adaptation in the event of time varying statistics. This method was tested on a mobile robot platform for the separate estimation of $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ [132], for estimating the model error for an autonomous helicopter [133], and for estimation of aerodynamic parameters from real flight data [134]. In all cases the performance of this adaptive UKF algorithm was found to give superior performance to that of the standard UKF in terms of convergence speed and estimation accuracy. The work in [132–134] is based around the UKF, however in this work we present a master-slave filtering framework based on the SR-UKF, referred to as MS-SRUKF, to take advantage of the improved numerical properties discussed previously. In this case, the slave filter adaptively tunes $\sqrt{\boldsymbol{R}_k^x}$ instead of $\boldsymbol{R}_k^x$.

The master filter can be derived by considering the general discrete-time non-linear system

$$\mathbf{x}_k = \mathbf{f}_x\left(\mathbf{x}_{k-1}\right) + \mathbf{w}_{x,k-1} \tag{5.64}$$

$$\mathbf{y}_k = \mathbf{h}_x\left(\mathbf{x}_k\right) + \mathbf{v}_{x,k} \tag{5.65}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state vector containing the structure parameters, $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement vector, and $\mathbf{w}_{x,k}$ and $\mathbf{v}_{x,k}$ are the process and measurement noise vectors, respectively. Since the state vector consists of only the structure parameters, and they have no process, Equation (5.64) could actually be replaced by the linear state equation

$$\mathbf{x}_k = \boldsymbol{F}_{x,k-1}\mathbf{x}_{k-1} + \mathbf{w}_{x,k-1} \tag{5.66}$$

where in this case $\boldsymbol{F}_{x,k} = \boldsymbol{I}, \quad \forall\, k$, where $\boldsymbol{I}$ is the identity matrix, and $\mathbf{w}_{x,k} = \mathbf{0}, \quad \forall\, k$. The measurement equation is non-linear in this application due to the non-linear camera model equation, therefore Equation (5.65) is as expressed above. Nevertheless, we shall proceed with the general case of non-linear equations for both the state and measurement equations in what follows.

If we assume that the measurement noise, $\mathbf{v}_{x,k}$, is Gaussian white, then the master filter measurement noise covariance matrix, $\boldsymbol{R}_k^x x$, will be a diagonal matrix, therefore the slave filter need only estimate the diagonal elements, which greatly simplifies the problem. If the diagonal elements of $\sqrt{\boldsymbol{R}_k^x}$ are denoted by the vector $\boldsymbol{\theta}_k \in \mathbb{R}^m$, and the measurement vector for the slave filter is denoted by $\boldsymbol{\phi}_k \in \mathbb{R}^m$, then the system dynamics for the slave filter can be represented by

$$\boldsymbol{\theta}_k = \mathbf{f}_\theta\left(\boldsymbol{\theta}_{k-1}\right) + \mathbf{w}_{\theta,k-1} \tag{5.67}$$

$$\boldsymbol{\phi}_k = \mathbf{g}\left(\boldsymbol{\theta}_k\right) + \mathbf{v}_{\theta,k} \tag{5.68}$$

where $\mathbf{w}_{\theta,k}$ and $\mathbf{v}_{\theta,k}$ are the process and measurement noise vectors for the slave filter, respectively. Equation (5.67) describes how the measurement noise variance changes with time as a consequence of a potentially changing external environment (i.e. due to transition from descent under a parachute to the jettisoning of the parachute and the transition from unpowered to powered descent, or atmospheric turbulence, varying thrust in powered descent, etc.). Although control signals (e.g. applied thrust) could be used to indicate that the noise environment has changed, precise modelling of the noise environment as a function of thrust and the characteristics of the rocket motor and spacecraft structure assembly would be required in order to handle these changes successfully. How exactly this would affect the feature tracking performance, through induced image blur, would be extremely difficult to model, and it would not cover the other types of disturbance, such as turbulence (the timing of which could not be predicted at all). Therefore, as far as the slave filter is concerned these changes can happen suddenly and without warning, so there is no suitable straightforward process model that can be defined to describe how the noise environment will change. We are therefore left with little option other than to assume that the process model is represented by the identity matrix multiplied by the previous state estimate, i.e. the state equation is the linear equation

$$\boldsymbol{\theta}_k = \boldsymbol{F}_{\theta,k-1}\boldsymbol{\theta}_{k-1} + \mathbf{w}_{\theta,k-1} \tag{5.69}$$

where $\boldsymbol{F}_{\theta,k} = \boldsymbol{I}, \quad \forall\, k$. However, as before, in what follows, the SR-UKF equations will be formulated as if the state equation is non-linear to maintain generality. In contrast to the master filter however, the process noise for the slave filter is not zero. Due to the inability to specify a suitable process model, any changes in the master filter measurement noise will have to be covered by a suitable addition of slave filter process noise. Unfortunately, the correct amount of process noise to be applied cannot be predicted and so an adaptive method, such as the ones described in Chapter 5 is adopted. Inspired by the success of the Mohamed and Schwarz method [111] applied to the UKF for the adaptive estimation of the measurement noise in [131], we choose to adopt this approach for the estimation of the process noise for the slave filter, where, obviously, instead of using Equation (5.54), we base our adaptive algorithm on the equation for adapting $\boldsymbol{Q}_k^\theta$:

$$\hat{\boldsymbol{Q}}_k^\theta = \boldsymbol{K}_{\theta,k}\hat{\boldsymbol{C}}_{\nu_{\theta,k}}\boldsymbol{K}_{\theta,k}^{\mathsf{T}}. \tag{5.70}$$

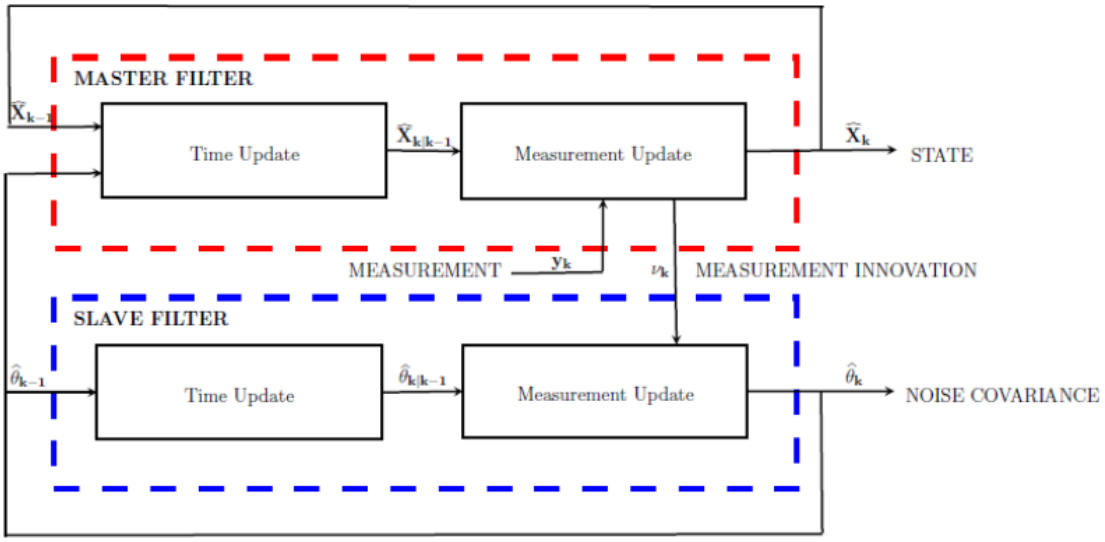


Figure 5.8: Schematic diagram of master-slave SR-UKF algorithm

The measurement model for the slave filter is derived from the predicted square-root measurement innovation covariance, ζ_k, of the master filter, and the measurement is given by

$$\boldsymbol{\phi}_k = diag\left\{\sqrt{\boldsymbol{\nu}_{\theta,k}\boldsymbol{\nu}_{\theta,k}^{\mathsf{T}}}\right\}. \tag{5.71}$$

A schematic illustrating the operation of the master-slave SR-UKF algorithm is presented in Figure 5.8. It is important to note that, even though this figure would seem to imply that the two filters operate in parallel to each other, the slave filter relies on the measurement innovation sequence of the master filter, therefore the slave filter cannot begin to process its measurements for the current time step until the master filter has completed its measurement updates. This means that very little, if any, increase in computational throughput can be realised by implementing parallel programming techniques on a global level. However, many of the component parts of this algorithm can be parallelised, such as computation of the sigma points, which can be computationally expensive when the number of tracked feature points is large.

The equations for each step in the Master-Slave Square-Root Unscented Kalman Filter algorithm are listed as follows:

MASTER FILTER

Initialisation

$$\hat{\mathbf{x}}_{0|0} = \mathrm{E}\left[\mathbf{x}_0\right] \tag{5.72}$$

$$\boldsymbol{S}^{x}_{0|0} = \text{chol}\left\{\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_{0|0}\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_{0|0}\right)^{\text{T}}\right]\right\} \tag{5.73}$$

Sigma Point Calculation

$$\boldsymbol{\chi}^{*}_{k-1|k-1} = \left[\hat{\mathbf{x}}_{k-1|k-1} \quad \hat{\mathbf{x}}_{k-1|k-1} + \eta^{x}\,\boldsymbol{S}^{x}_{k-1|k-1} \quad \hat{\mathbf{x}}_{k-1|k-1} - \eta^{x}\,\boldsymbol{S}^{x}_{k-1|k-1}\right] \tag{5.74}$$

Time Update

$$\boldsymbol{\chi}^{*}_{k|k-1} = f_{x}(\boldsymbol{\chi}^{*}_{k-1|k-1}) \tag{5.75}$$

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2N} W^{m}_{x,i}\,\boldsymbol{\chi}^{*}_{i,k|k-1} \tag{5.76}$$

$$\boldsymbol{S}^{x-}_{k|k-1} = \text{qr}\left\{\left[\sqrt{W^{c}_{x,1}}\left(\boldsymbol{\chi}^{*}_{1:2N,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right) \quad \sqrt{\hat{\boldsymbol{Q}}^{x}_{k-1|k-1}}\right]\right\} \tag{5.77}$$

$$\boldsymbol{S}^{x}_{k|k-1} = \text{cholupdate}\left\{\boldsymbol{S}^{x-}_{k|k-1},\ \left(\boldsymbol{\chi}^{*}_{0,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right),\ W^{c}_{x,0}\right\} \tag{5.78}$$

Recalculate Sigma Points

$$\boldsymbol{\chi}_{k|k-1} = \left[\hat{\mathbf{x}}_{k|k-1} \quad \hat{\mathbf{x}}_{k|k-1} + \eta^{x}\,\boldsymbol{S}^{x}_{k|k-1} \quad \hat{\mathbf{x}}_{k|k-1} - \eta^{x}\,\boldsymbol{S}^{x}_{k|k-1}\right] \tag{5.79}$$

Measurement Prediction

$$\boldsymbol{\gamma}_{k|k-1} = h_{x}(\boldsymbol{\chi}_{k|k-1}) \tag{5.80}$$

$$\hat{\mathbf{y}}_{k|k-1} = \sum_{i=0}^{2N} W^{m}_{x,i}\,\boldsymbol{\gamma}_{i,k|k-1} \tag{5.81}$$

Measurement Update

$$\boldsymbol{S}^{-}_{y_k} = \text{qr}\left\{\left[\sqrt{W^{c}_{x,1}}\left(\boldsymbol{\gamma}_{1:2N,k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right) \quad \sqrt{\hat{\boldsymbol{R}}^{x}_{k-1|k-1}}\right]\right\} \tag{5.82}$$

$$\boldsymbol{S}_{y_k} = \text{cholupdate}\left\{\boldsymbol{S}^{-}_{y_k},\ \left(\boldsymbol{\gamma}_{0,k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right),\ W^{c}_{x,0}\right\} \tag{5.83}$$

$$\boldsymbol{P}_{x_k y_k} = \sum_{i=0}^{2N} W_{x,i}^{c} \left(\boldsymbol{\chi}_{i,k|k-1} - \hat{\mathbf{x}}_{k|k-1} \right) \left(\boldsymbol{\gamma}_{i,k|-1} - \hat{\mathbf{y}}_{k|k-1} \right)^{\mathrm{T}} \tag{5.84}$$

$$\boldsymbol{K}_{k}^{x} = \left(\boldsymbol{P}_{x_k y_k} / \boldsymbol{S}_{y_k}^{\mathrm{T}} \right) / \boldsymbol{S}_{y_k} \tag{5.85}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \boldsymbol{K}_{k}^{x} \left(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1} \right) \tag{5.86}$$

$$\boldsymbol{S}_{k|k}^{x} = \mathrm{cholupdate} \left\{ \boldsymbol{S}_{k|k-1}^{x}, \ \boldsymbol{K}_{k}^{x} \boldsymbol{S}_{y_k}, \ -1 \right\} \tag{5.87}$$

where

$$W_{x,0}^{m} = \frac{\lambda_x}{N + \lambda_x}$$

$$W_{x,0}^{c} = \frac{\lambda_x}{N + \lambda_x} + \left(1 - \alpha_x^2 + \beta_x \right)$$

$$W_{x,i}^{m} = W_{x,i}^{c} = \frac{1}{2(N + \lambda_x)}$$

$$\lambda_x = \alpha_x^2 \left(N + \kappa_x \right) - N$$

$$\eta^x = \sqrt{N + \lambda_x}$$

$N =$ number of states in master filter

$0.0001 \leq \alpha_x \leq 1$

$\beta_x = 2$ (optimal for Gaussian distribution)

$\kappa_x = 0$ (usual for state estimation)

SLAVE FILTER

Initialisation

$$\hat{\theta}_{0|0} = \mathrm{E}\left[\theta_0\right] \tag{5.88}$$

$$\boldsymbol{S}^{\theta}_{0|0} = \mathrm{chol}\left\{\mathrm{E}\left[\left(\theta_0 - \hat{\theta}_{0|0}\right)\left(\theta_0 - \hat{\theta}_{0|0}\right)^{\mathrm{T}}\right]\right\} \tag{5.89}$$

Sigma Point Calculation

$$\boldsymbol{\vartheta}_{k-1|k-1} = \begin{bmatrix}\hat{\theta}_{k-1|k-1} & \hat{\theta}_{k-1|k-1} + \eta^{\theta}\boldsymbol{S}_{k-1|k-1} & \hat{\theta}_{k-1|k-1} - \eta^{\theta}\boldsymbol{S}_{k-1|k-1}\end{bmatrix} \tag{5.90}$$

Time Update

$$\boldsymbol{\vartheta}^{*}_{k|k-1} = f_{\theta}(\boldsymbol{\vartheta}_{k-1|k-1}) \tag{5.91}$$

$$\hat{\theta}_{k|k-1} = \sum_{i=0}^{2M} W^{m}_{\theta,i}\,\boldsymbol{\vartheta}^{*}_{i,k|k-1} \tag{5.92}$$

$$\boldsymbol{S}^{\theta-}_{k|k-1} = \mathrm{qr}\left\{\begin{bmatrix}\sqrt{W^{c}_{\theta,1}}\left(\boldsymbol{\vartheta}^{*}_{1:2M,k|k-1} - \hat{\theta}_{k|k-1}\right) & \sqrt{Q^{\theta}_{k-1|k-1}}\end{bmatrix}\right\} \tag{5.93}$$

$$\boldsymbol{S}^{\theta}_{k|k-1} = \mathrm{cholupdate}\left\{\boldsymbol{S}^{\theta-}_{k|k-1},\ \left(\boldsymbol{\vartheta}^{*}_{0,k|k-1} - \hat{\theta}_{k|k-1}\right),\ W^{c}_{\theta,0}\right\} \tag{5.94}$$

Recalculate Sigma Points

$$\boldsymbol{\vartheta}_{k|k-1} = \begin{bmatrix}\hat{\theta}_{k|k-1} & \hat{\theta}_{k|k-1} + \eta^{\theta}\boldsymbol{S}^{\theta}_{k|k-1} & \hat{\theta}_{k|k-1} - \eta^{\theta}\boldsymbol{S}^{\theta}_{k|k-1}\end{bmatrix} \tag{5.95}$$

Measurement Prediction

$$\varsigma_{k|k-1} = h_{\theta}(\boldsymbol{\vartheta}_{k|k-1}) \tag{5.96}$$

$$\hat{\mathrm{s}}_{k|k-1} = \sum_{i=0}^{2M} W^{m}_{\theta,i}\,\varsigma_{k|k-1} \tag{5.97}$$

Measurement Update

$$\boldsymbol{S}_{s_k}^{-} = \mathrm{qr}\left\{\left[\sqrt{W_{\theta,1}^{c}}\left(\boldsymbol{\varsigma}_{1:2M,k|k-1} - \hat{\mathbf{s}}_{k|k-1}\right) \quad \sqrt{R_{k-1|k-1}^{\theta}}\right]\right\} \tag{5.98}$$

$$\boldsymbol{S}_{s_k} = \mathrm{cholupdate}\left\{\boldsymbol{S}_{s_k}^{-},\ \left(\boldsymbol{\varsigma}_{0,k|k-1} - \hat{\mathbf{s}}_{k|k-1}\right),\ W_{\theta,0}^{c}\right\} \tag{5.99}$$

$$\boldsymbol{P}_{\theta_k s_k} = \sum_{i=0}^{2M} W_{\theta,i}^{c}\left(\boldsymbol{\vartheta}_{i,k|k-1} - \hat{\boldsymbol{\theta}}_{k|k-1}\right)\left(\boldsymbol{\varsigma}_{i,k|k-1} - \hat{\mathbf{s}}_{k|k-1}\right)^{\mathrm{T}} \tag{5.100}$$

$$\boldsymbol{K}_{k}^{\theta} = \left(\boldsymbol{P}_{\theta_k s_k} / \boldsymbol{S}_{s_k}^{\mathrm{T}}\right) / \boldsymbol{S}_{s_k} \tag{5.101}$$

$$\hat{\boldsymbol{\theta}}_{k|k} = \hat{\boldsymbol{\theta}}_{k|k-1} + \boldsymbol{K}_{k}^{\theta}\left(\mathbf{s}_k - \hat{\mathbf{s}}_{k|k-1}\right) \tag{5.102}$$

$$\boldsymbol{S}_{k|k}^{\theta} = \mathrm{cholupdate}\left\{\boldsymbol{S}_{k|k-1}^{\theta},\ \boldsymbol{K}_{k}^{\theta}\boldsymbol{S}_{s_k},\ -1\right\} \tag{5.103}$$

where

$$W_{\theta,0}^{m} = \frac{\lambda_\theta}{M+\lambda_\theta}$$

$$W_{\theta,0}^{c} = \frac{\lambda_\theta}{M+\lambda_\theta} + \left(1 - \alpha_\theta^2 + \beta_\theta\right)$$

$$W_{\theta,i}^{m} = W_{\theta,i}^{c} = \frac{1}{2(M+\lambda_\theta)}$$

$$\lambda_\theta = \alpha_\theta^2\left(M + \kappa_\theta\right) - M$$

$$\eta^{\theta} = \sqrt{M+\lambda_\theta}$$

$M =$ number of states in slave filter

$0.0001 \leq \alpha_\theta \leq 1$

$\beta_\theta = 2$ (optimal for Gaussian distribution)

$\kappa_\theta = 0$ (usual for state estimation)

where the measurement model equations for the slave filter are derived from the measurement innovation equations of the master filter (Equations (5.82) & (5.83)), i.e.

$$\boldsymbol{\varsigma}^{-}_{k|k-1} = h_{\theta}(\boldsymbol{\vartheta}_{k|k-1})^{-} = \mathrm{qr}\left\{\left[\sqrt{W^{c}_{x,1}}\left(\boldsymbol{\gamma}_{1:2N,k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right) \quad \sqrt{\hat{\boldsymbol{R}}^{x}_{k|k-1}}\right]\right\} \tag{5.104}$$

$$\boldsymbol{\varsigma}_{k|k-1} = h_{\theta}(\boldsymbol{\vartheta}_{k|k-1}) = \mathrm{cholupdate}\left\{\boldsymbol{\varsigma}^{-}_{k|k-1}, \ \left(\boldsymbol{\gamma}_{0,k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right), \ W^{c}_{x,0}\right\} \tag{5.105}$$

Some proof of concept results using this master-slave configuration on a similar dataset to those used in Chapter 4 will be presented in Section 5.8. However, before we can do this, it is important to note that, although the master filter can be adaptively tuned by the slave filter, the slave filter itself needs to be adequately tuned to ensure that it does not corrupt the performance of the master filter. Thus, aside from the ability for the master filter to adapt to a potentially changing external noise environment, in terms of actually tuning the filter, all that has been achieved is an offsetting of the tuning to the slave filter instead of the master filter itself, as would be the case in a conventional filtering framework. A traditional manual tuning could be carried out using a trial and error approach, however the drawback of this is that it is impossible to be certain that the most optimal set of tuning parameters has been found, even if the filter can be seen to be performing adequately by comparison to a ground truth. A more sophisticated approach to this can, and should be implemented using a mathematical optimality criterion. This is the subject of the following section, which makes use of a particle swarm optimisation technique that aims to optimally initialise the two filters by finding the most suitable set of initial tuning parameters. Once these initial parameters have been identified, the slave filter can then further refine the tuning parameters of the master filter (if necessary) and make any required adaptations to any changes in the external noise environment.

5.7 Particle Swarm Optimisation Filter Initialisation

In order for the adaptive techniques described in Section 5.5 to work effectively, a good initial guess of the tuning parameters is required. This is due to the fact that none of the adaptive algorithms discussed in Section 5.5 are able to completely remove the need for *a priori* stochastic information, thus the filters still require initial values for the measurement and process noise covariance and the state error covariance [115]. Furthermore, these initial estimates still need to be reasonable due to the delay in adaptation as a consequence of having to accumulate N noise samples before the first adaptive calculation can be carried out. While this argument applies more directly to the slave filter due to the particular adaptive mechanism employed in that filter, the master filter must also be performing reasonably during this time in order for the slave filter to adapt appropriately to the incoming measurements. This chapter presents a method by which a reasonable initial set of tuning parameters may be obtained by using the

measurement information through a technique known as particle swarm optimisation. The additional adaptive mechanisms incorporated into the filter design may then further refine these initial parameters in order to increase robustness to a potentially varying noise environment and improve the overall accuracy of the estimated scene structure.

5.7.1 Particle Swarm Optimisation

Particle swarm optimisation (PSO) was originally devised by Kennedy and Eberhart [135, 136] as a computational method for the optimisation of continuous non-linear functions. The technique was discovered via simulations of the simplified social behaviour involved in the motion patterns of bird flocking, fish schooling and swarming insects. This motion is typically characterised by synchronised and seemingly choreographed sudden changes of direction, scattering and regrouping, as well as the non-existence of collisions. In addition to the highly synchronised motion, it has been theorised that individual members of the group can profit from the discoveries and previous experience of all other members during the search for food [135], which enables the flock or swarm to converge on rich sources of food. Clearly, it would seem that some form of intercommunication is involved in order to enable such group behaviour. By modelling this intercommunication and motion characteristics Kennedy and Eberhart were able to devise a technique that is capable of converging on the optimum solution of a non-linear equation.

In PSO the location of each particle inside a search space of possible solutions is regarded as a candidate solution of the problem. To simulate the social behaviour, each particle maintains a record of its previous best location, as well as having access to the global best location found by the entire swarm, both of which are found by evaluating a cost function for the particular non-liner equation that is being solved. The motion of each particle is iteratively updated according to the combined influence of a random parameter and the two measures of "best" location, which over time has the effect of driving the swarm closer to the optimum solution. The algorithm is stopped once convergence has occurred or after a predefined maximum number of iterations. More formally, the particle position is updated according to the following equation:

$$x_i\left(t+1\right) = x_i\left(t\right) + v_i\left(t+1\right) \tag{5.106}$$

where

$$v_i\left(t+1\right) = wv_i\left(t\right) + c_1 r_1\left[p_i - x_i(t)\right] + c_2 r_2\left[p_g - x_i(t)\right] \tag{5.107}$$

is the i^{th} particle velocity at time t, p_i is the personal best location of the i^{th} particle, p_g is the global best location found by the swarm, w is an inertia weight governing the influence of velocity from the previous iteration, c_1 and c_2 are known as acceleration coefficients, which weight the influence of the terms involving the current best known locations, and r_1 and r_2 are uniformly distributed random numbers between 0 and 1 [137]. One choice for the value of the inertia weight, w is to compute it as a linearly decreasing value between upper and lower bounds so that in the beginning the emphasis is more on searching the solution space, whereas near the end the focus is on

convergence since the relative importance of the influence of p_i and p_g is increased. This approach was adopted in [138], where the inertia weight was calculated in each iteration by

$$w = w_{max} - \frac{t\,(w_{max} - w_{min})}{t_{max}} \tag{5.108}$$

where t is the current iteration, t_{max} is the maximum allowed number of iterations, w_{max} is the maximum inertia weight and is usually set to 0.9, and w_{min} is the minimum inertia weigh and is usually set to 0.4. Appropriate values for the acceleration coefficients, c_1 and c_2 are 2 as used in [135], or 2.05 as used in [137].

5.7.2 Application to Filter Initialisation

Particle swarm optimisation was first incorporated to assist in adaptive tuning of Kalman filter-based algorithms by Jwo and Chang [139], and shortly after, and seemingly independently, in a number of papers by Jatoth et al [137, 140–143]. Applying PSO to adaptive Kalman filter tuning is relatively straightforward, the only difficulty is in determining an appropriate cost/fitness function once the parameters that need to be identified have been established.

Jwo and Chang [139] took inspiration from covariance matching and process noise covariance scaling in order to derive a cost function of the form

$$J = \frac{trace\left\{\hat{\boldsymbol{C}}_{\nu_k}\right\}}{trace\left\{\boldsymbol{H}_k\left(\boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_{k-1}^{\mathsf{T}} + \lambda_k\boldsymbol{Q}_{k-1}\right)\boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k\right\}} \tag{5.109}$$

where λ_k is a scale factor to be determined that increases or decreases the influence of the previous process noise covariance, $\boldsymbol{Q}_{k-1}$, in the calculation of the Kalman gain (via $\boldsymbol{P}_{k|k-1}$) for the current time step, $\boldsymbol{K}_k$, based on the measurement innovation sequence. Note that $\hat{\boldsymbol{C}}_{\nu_k}$ is calculated as before (see Equation (5.41)). The goal is to use PSO to find a solution for how the previous process noise covariance matrix should be scaled to make the predicted measurement innovation covariance (denominator) consistent with the actual covariance (numerator). Thus the goal is to obtain $J \approx 1$. After convergence the scale factor can be determined and the process noise covariance matrix to be used in the next time step is given by

$$\boldsymbol{Q}_k = \lambda_k\boldsymbol{Q}_{k-1}. \tag{5.110}$$

The PSO algorithm can be computationally demanding especially when the dimension of the search space is large. Therefore to avoid unnecessary computation, the PSO adaptation was only executed in [139] when the cost function, using $\lambda_k = 1$, deviates significantly from 1. Significant improvements in the estimation results in a GPS navigation application were observed in [139] using this technique, compared to standard Kalman filtering.

Ramakoti et al [140] apply the PSO algorithm to tune a Kalman filter that is used to track a moving ball in a sequence of images, with their results showing improved

tracking performance over the conventional Kalman filter. In [141] the PSO tuning algorithm is applied to an extended Kalman filtering problem for tracking a manoeuvring target, where again improved performance was obtained using a simulated moving target. References [137, 142] apply the PSO tuning algorithm to an unscented Kalman filter for tracking aircraft from bearings-only radar measurements, showing that improved tracking results can also be obtained compared to the standard UKF. Finally, in [143] the PSO assisted UKF was applied to tracking objects undergoing a ballistic atmospheric re-entry. Once again improved results were obtained from the PSO-UKF compared to the UKF. In all of these works, which we refer to collectively as Jatoth et al, the PSO algorithm used a fitness function based on an approximate deterministic function for relating the tuning parameters to a mean-square error criterion of optimisation. However, no details were given as to the specific form of fitness function that was used.

Panigrahi et al [144] applied an adaptive UKF for the estimation of voltage flicker and harmonics in power networks. In addition to the adaptive mechanism, a PSO algorithm, with inertia weights adaptively determined using fuzzy logic, was used to obtain approximately optimal initial values for the UKF tuning parameters. Any subsequent changes in the noise environment were handled by the adaptive mechanism in the UKF. A fitness function was specified for the PSO in terms of an objective function, however no details about the form of the objective function were given.

In our work, we adopt an approach inspired by Panigrahi et al [144], in that we are employing PSO as a means of obtaining a suitable set of initial values for the tuning parameters. Since we have two filters to initialise, we must apply PSO to both the master filter and the slave filter. For the master filter the parameters of interest are the initial state vector $\hat{\mathbf{x}}_{0|0}$, the initial measurement noise covariance $\boldsymbol{R}_0^x$, the initial state error covariance $\boldsymbol{P}_{0|0}^x$, and the SR-UKF parameters α^x and β^x. For the slave filter the parameters of interest are the process noise covariance $\boldsymbol{Q}_0^\theta$, the initial state error covariance $\boldsymbol{P}_{0|0}^\theta$, and the SR-UKF parameters α^θ and β^θ. Under the assumption of Gaussian white noise, the covariance matrices will be diagonal, therefore we need only concern ourselves with the search for suitable diagonal elements for these matrices, which reduces the dimension of the search-space considerably. The location of the i^{th} particle within the search space at iteration t is therefore specified by the following vector:

$$x_i(t) = \begin{bmatrix} \alpha_{i,t}^x \\ \beta_{i,t}^x \\ \alpha_{i,t}^\theta \\ \beta_{i,t}^\theta \\ \hat{\mathbf{x}}_{0|0,i,t} \\ diag\left\{\boldsymbol{P}_{0|0,i,t}^x\right\} \\ diag\left\{\boldsymbol{R}_{0,i,t}^x\right\} \\ diag\left\{\boldsymbol{P}_{0|0,i,t}^\theta\right\} \\ diag\left\{\boldsymbol{Q}_{0,i,t}^\theta\right\} \end{bmatrix}. \tag{5.111}$$

Since the goal of this initialisation is to estimate suitable initial tuning parameters to be used until sufficient noise samples have been accumulated for the slave filter adaptive mechanism to begin adaptively tuning $\boldsymbol{Q}_k^{\theta}$, the adaptive mechanism for the slave filter will be inactive during PSO initialisation. However, we will be allowing the slave filter to update the measurement noise covariance of the master filter during the PSO initialisation, which will be carried out over the first N image frames in order to ensure that the tuning parameters will be valid for a reasonable length of time — if only the first pair of images was used it could be possible to obtain a set of parameters that are only valid for that pair of images.

For each particle, a fitness function will be evaluated in each filter after the *a posteriori* state estimates have been calculated for the N^{th} image. The fitness function that will be used in the master filter is derived based on the adaptive technique used by Zhou et al [131] for their adaptive UKF. In [131], the equation for adapting the measurement noise covariance matrix was given as

$$\hat{\boldsymbol{R}}_k = \hat{\boldsymbol{C}}_{\nu_k}^{+} + \boldsymbol{P}_{y_k y_k}^{+} \tag{5.112}$$

where

$$\boldsymbol{P}_{y_k y_k}^{+} = \sum_{i=0}^{2n} W_i^c \left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k,i}\right) - \hat{\mathbf{y}}_{k|k}\right]\left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k,i}\right) - \hat{\mathbf{y}}_{k|k}\right]^{\mathsf{T}} \tag{5.113}$$

where the $\hat{\mathbf{x}}_{k|k,i}$ represent the sigma points drawn from the UKF a posteriori estimates, i.e.

$$\hat{\mathbf{x}}_{k|k,0} = \hat{\mathbf{x}}_{k|k} \tag{5.114}$$

$$\hat{\mathbf{x}}_{k|k,i} = \hat{\mathbf{x}}_{k|k} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k}}\right) \tag{5.115}$$

$$\hat{\mathbf{x}}_{k|k,i+n} = \hat{\mathbf{x}}_{k|k} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k}}\right) \tag{5.116}$$

$$i = 1, \cdots, n.$$

and where

$$\hat{\boldsymbol{C}}_{\nu_k}^{+} = \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{\nu}_j \boldsymbol{\nu}_j^{\mathsf{T}} \tag{5.117}$$

where

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k}\right). \tag{5.118}$$

Equation (5.112) is derived by recognising that the *a posteriori* theoretical measurement innovation covariance, $\boldsymbol{C}_{\nu_k}^{+}$, is given by

$$\boldsymbol{C}_{\nu_k}^{+} = \boldsymbol{R}_k - \boldsymbol{P}_{y_k y_k}^{+} \tag{5.119}$$

and then applying the covariance matching principle of setting this equation equal to Equation (5.117). Note that since covariance matrices are at least positive semi-definite by definition, Equation (5.112) implies that $\boldsymbol{R}_k \geq \boldsymbol{P}^+_{y_k y_k}$, therefore $\boldsymbol{C}_{\nu_k}$ will always be at least positive semi-definite when calculated using Equation (5.119), despite the subtraction operation. Taking inspiration from [139], we therefore formulate the following cost function to be used in the master filter:

$$J_x = abs\left(1 - \frac{trace\left\{\hat{\boldsymbol{C}}^+_{\nu_k}\right\}}{trace\left\{\hat{\boldsymbol{R}}_k - \boldsymbol{P}^+_{y_k y_k}\right\}}\right), \tag{5.120}$$

which equals zero when the two measurement innovation covariances match.

This cost function cannot be used in the slave filter, because for the slave filter we have made the assumption that the measurement noise covariance is zero, which according to Equation (5.112), implies $\boldsymbol{C}^+_{\nu_k}$ and $\boldsymbol{P}^+_{y_k y_k}$ must also be zero due to the requirement for covariance matrices to be at least positive semi-definite. To derive a cost function for the slave filter we must consider the equation for the measurement innovation covariance using a priori information, which from [111], is given by

$$\boldsymbol{C}^-_{\nu_k} = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^{\mathsf{T}} + \boldsymbol{R}_k \tag{5.121}$$

By making a similar substitution to that made by Zhou et al [131], we can re-write this equation for the UKF:

$$\boldsymbol{C}^-_{\nu_k} = \boldsymbol{P}^-_{y_k y_k} + \boldsymbol{R}_k \tag{5.122}$$

where

$$\boldsymbol{P}^-_{y_k y_k} = \sum_{i=0}^{2n} W_i^c \left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1,i}\right) - \hat{\mathbf{y}}_{k|k-1}\right]\left[\mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1,i}\right) - \hat{\mathbf{y}}_{k|k-1}\right]^{\mathsf{T}} + \boldsymbol{R}_k \tag{5.123}$$

where

$$\hat{\mathbf{y}}_{k|k-1} = \sum_{i=0}^{2n} W_i^m \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1,i}\right), \tag{5.124}$$

where the $\hat{\mathbf{x}}_{k|k-1,i}$ represent the sigma points drawn from the UKF *a priori* estimates, i.e.

$$\hat{\mathbf{x}}_{k|k-1,0} = \hat{\mathbf{x}}_{k|k-1} \tag{5.125}$$

$$\hat{\mathbf{x}}_{k|k-1,i} = \hat{\mathbf{x}}_{k|k-1} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k-1}}\right) \tag{5.126}$$

$$\hat{\mathbf{x}}_{k|k-1,i+n} = \hat{\mathbf{x}}_{k|k-1} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k|k-1}}\right) \tag{5.127}$$

$$i = 1, \cdots, n.$$

and where $\boldsymbol{P}_{k|k-1}$ is given by

$$\boldsymbol{P}_{k|k-1} = \sum_{i=0}^{2n} W_i^c \left[\mathbf{f}_k\left(\hat{\mathbf{x}}^*_{k|k-1,i}\right) - \hat{\mathbf{x}}_{k|k-1}\right]\left[\mathbf{f}_k\left(\hat{\mathbf{x}}^*_{k|k-1,i}\right) - \hat{\mathbf{x}}_{k|k-1}\right]^{\mathsf{T}} + \boldsymbol{Q}_{k-1}, \quad (5.128)$$

which explicitly establishes the connection to $\boldsymbol{Q}_k$. In this case the $\hat{\mathbf{x}}^*_{k|k-1,i}$ represent the sigma points drawn from the UKF *a posteriori* estimates from the previous time step, i.e.

$$\hat{\mathbf{x}}^*_{k|k-1,0} = \hat{\mathbf{x}}_{k-1|k-1} \quad (5.129)$$

$$\hat{\mathbf{x}}^*_{k|k-1,i} = \hat{\mathbf{x}}_{k-1|k-1} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k-1|k-1}}\right) \quad (5.130)$$

$$\hat{\mathbf{x}}^*_{k|k-1,i+n} = \hat{\mathbf{x}}_{k-1|k-1} + \left(\sqrt{(n+\lambda)\boldsymbol{P}_{k-1|k-1}}\right) \quad (5.131)$$

$$i = 1, \cdots, n.$$

Therefore, the cost function for the slave filter is given by

$$J_\theta = abs\left(1 - \frac{trace\left\{\hat{\boldsymbol{C}}^-_{\nu_k}\right\}}{trace\left\{\boldsymbol{P}^-_{y_k y_k}\right\}}\right) \quad (5.132)$$

where $\hat{\boldsymbol{C}}^-_{\nu_k}$ is given by

$$\hat{\boldsymbol{C}}^-_{\nu_k} = \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{\nu}_j \boldsymbol{\nu}_j^{\mathsf{T}} \quad (5.133)$$

where

$$\boldsymbol{\nu}_k = \mathbf{y}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1}\right). \quad (5.134)$$

Finally, we must take into account the particle fitness due to the combined effect of the two filters to ensure a set of tuning parameters are found for each filter that are consistent with each other. This is easily accomplished by adding the two individual costs:

$$J_{tot} = J_x + J_\theta. \quad (5.135)$$

The goal of the PSO algorithm is therefore to minimise Equation (5.135).

5.8 Proof of Concept Results

To assess the performance of the proposed approach a simple test scenario was used based on artificial images produced using the Planet and Asteroid Natural scene Generation Utility (PANGU), which is a software package that allows the creation of

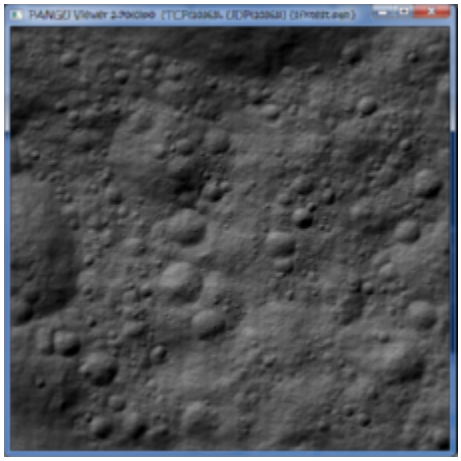


Figure 5.9: Example PANGU image used in PSO initialised SR-UKF, at an altitude of 2000m, with a nadir viewing direction.

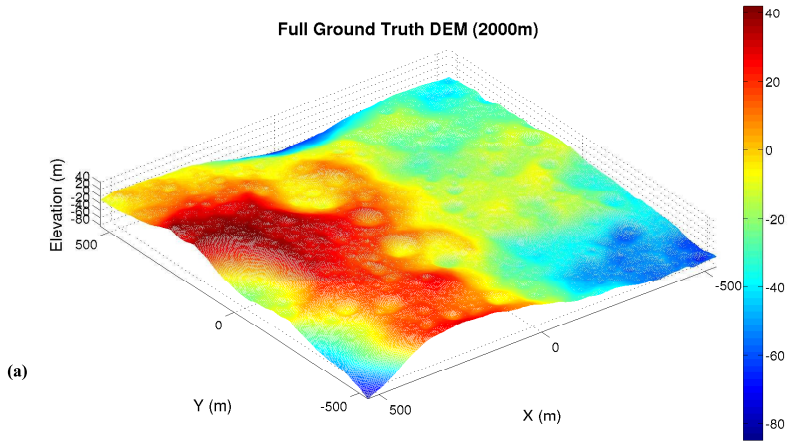


Figure 5.10: Dense ground truth DEM of the terrain visible in the first image (from 2000m altitude) of the sequence used in the PSO initialised SR-UKF.

realistic 3D models of planetary surfaces, as was used in Chapter 4. An example image is shown in Figure 5.9. Therefore the image sequence represents a physically realistic portion of a descent towards the surface. The image sequence begins at an altitude of 2000m and consists of 100 images at 20 frames per second, with the spacecraft descending in the nadir direction at a constant speed of 100m/s. Each image consists of 512x512 pixels covering a 30x30 degree field of view, with the bore-sight direction corresponding to the view direction of the centre pixel $(256.5, 256.5)$. With a focal length of 30mm, at 2000m altitude, each pixel in the image corresponds to approximately 2m on the surface. A ground truth for the shape of the terrain was obtained using the PANGU LIDAR tool to provide a dense 3D digital elevation model, which is shown in Figure 5.10. Since the tracked feature points will be sparse in comparison to the full ground truth

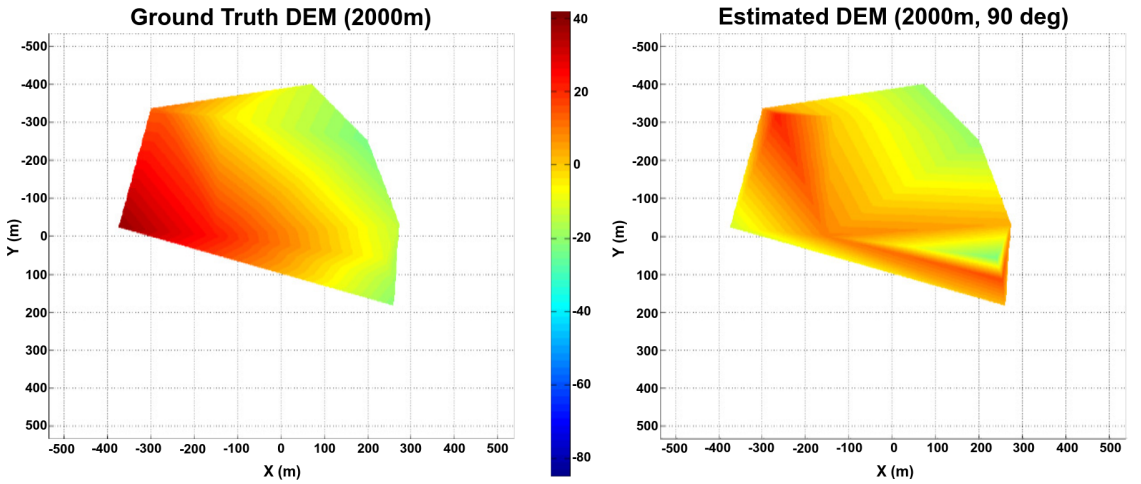


Figure 5.11: PSO initialised SR-UKF structure estimation results: Sparse ground truth DEM (left), Estimated DEM (right).

DEM, to provide a straight forward means of comparison between the estimated structure and the ground truth, we will artificially reduce the density of the ground truth DEM by selecting points that correspond to the tracked feature points. Both the sparse ground truth points and the estimated structure points will then be used to construct a DEM by fitting a dense mesh-grid and using linear interpolation between the points. Before this can be done, it must be noted that the SFM algorithm estimates the distance of the feature points from the point of view of the world frame, therefore we must subtract these values from a ground reference plane so that the height of the points with respect to this plane can be plotted. For this we use a reference depth corresponding to the initial altitude of the spacecraft, i.e. 2000m. The resulting DEMs are presented in Figure 5.11, which shows the sparse ground truth on the left and the estimated DEM on the right.

From Figure 5.11 it can be seen that the estimated DEM is in approximate agreement with the sparse ground truth DEM in that the terrain can be seen to increase in elevation from around -20m on the right to around +20m on the left. This can be seen most clearly at the topmost part of the figure where the best case error is very small, perhaps in the order of around 1m or better. The accuracy decreases slightly, but overall remains promising down to around the +100m line in the Y direction. In the bottom 200–300m of the coloured region this agreement breaks down to an extent since some anomalies in the estimated results can clearly be seen. In the extreme bottom-right corner of the coloured region the terrain height is in error by around 35m, and just above this is what appears to be an anomalous green triangle, but in fact this is a region of relatively low error for the most part, compared to the surrounding area. The worst case error can be seen in the extreme bottom-left corner of the coloured region where the error magnitude is approximately 50m. However, despite these obvious anomalies, we can consider the results to be promising considering that this simple test case in fact represents a particularly challenging scenario due to the high altitude of the spacecraft. Improvements in the structure estimates should occur naturally as the distance between the surface and the spacecraft decreases.

It should also be noted that the resulting DEM is quite small in area and very sparse. The white region surrounding the coloured area represents the part of the terrain for which no data is available. This is because the tracked feature points were clustered around this central part of the image since it is the most textured region. This is largely coincidental since the terrain is generated randomly in PANGU. The sparsity of the estimated DEM on the other hand is wholly due to a large number of feature points being lost during the tracking process. This was observed to be due to divergence in the iterative minimisation step in the KLT tracking algorithm. If the tracking solution for a particular feature does not converge after a certain number of iterations divergence is assumed and that feature is rejected.

5.8.1 Conclusions

An adaptive, self tuning, filtering framework has been developed using a parallel master-slave configuration based on the square-root unscented Kalman filter to estimate the structure of the terrain surrounding a potential landing site from a short sequence of images representing a portion of a descent. This technique has the capability of adapting to changing noise statistics, which can be expected to occur during the descent phase of planetary landing. Some promising initial results have been obtained that suggests that, with further development, this method may have the potential to supply the required accuracy for hazard detection in support of a pin-point landing for future planetary surface exploration missions. This is by far the most significant contribution in this thesis and it represents a great deal of the development effort in this work. As such it will be more extensively tested in the remaining chapters.

While overall the results showed promising performance, some anomalies were observed in the estimated terrain structure. There are a number of possible reasons for this. Firstly, the feature tracking method used in the production of these results (conventional KLT) may not be capable of tracking the features to a high enough accuracy. KLT type methods are known to drift from the true feature point over time, thus potentially introducing significant errors in the structure estimation, as mentioned previously. In the remaining chapters the algorithm will be tested in conjunction with the more robust TR-KLT feature tracker in an attempt to reduce the accumulation of errors in the feature measurements. Secondly, although the slave filter is responsible for adaptively tuning the master filter, the slave filter (in these results) still requires manual tuning. The process of manually tuning filters consists of a long iterative process of trial and error and, as such, it is very unlikely that an optimal set of tuning parameters will be obtained. Incorrect tuning parameters can seriously degrade the performance of filtering algorithms. In the remaining chapters, the algorithm is tested with additional adaptive mechanisms applied to the slave filter (covariance matching) thus completing the fully adaptive system.

6 VISILAB DATASET ACQUISITION

This chapter describes the set-up and utilisation of a computer vision test bench, located at ESA-ESTEC in The Netherlands, in order to generate a set of realistic descent trajectories, using real camera hardware in conjunction with a scaled mock-up of a portion of the Lunar south pole region. The chapter begins with an overview of the VISILAB equipment, which includes a discussion of the decisions that were made during the design of the test bench, the details of the camera system, how the surface mock-up was manufactured, details of the design of the computer controlled camera mounting system, important coordinate reference frames, and a description of the illumination system. Following this, a brief discussion of the changes that had to be made in order for the VISILAB test bench to be used in this project is given, including details of the different configuration options that were identified. The next section briefly discusses camera calibration and presents a procedure for the calibration of the VISILAB camera system along with a presentation of the calibration results obtained using this procedure. The final section then describes the details behind the calculation/modelling of a representative descent trajectory for a Lunar type planetary landing mission, a method of scaling this trajectory to the VISILAB test bench is then described using the previously obtained calibration parameters, and finally the details of the various recorded datasets is given. In addition to the collection of representative datasets, the primary contribution of this chapter is the testing of the fully adaptive MS-SRUKF algorithm on real camera images, thus achieving a technology readiness level of 3-4. The results demonstrate that reasonably good levels of accuracy can be achieved using the developed SFM algorithm and filtering framework.

6.1 VISILAB Overview

VISILAB is a computer vision-based navigation test bench at ESA-ESTEC in The Netherlands. It consists of a scaled mock-up of a portion of the Lunar south polar region, an illumination system, and a camera mounted on a motorised 2-axis linear table. The purpose of this equipment is to enable the testing of vision-based navigation and hazard detection algorithms, with camera hardware-in-the-loop, using representative terrain images and precisely controlled trajectories under a simulated entry, descent and landing scenario.

VISILAB was originally designed during the course of a previous PhD project. This previous project focussed on the development and testing of an absolute visual navigation system known as LION (Landing Inertial and Optical Navigation) that aimed to provide absolute position and motion estimation, using digital images during a lunar descent, with sufficient accuracy to enable a pin-point landing operation to be carried out. The specific descent scenario investigated for this project was that of a Lunar sample return mission at the lunar south pole. Thus VISILAB was very much designed with this in

mind, to the point where the terrain surface was chosen to be a scaled mock-up of a portion of the Lunar south polar region, based on altimetry data obtained from the NASA Lunar Reconnaissance Orbiter. Although the VISILAB test bench was designed during the course of this project, with that project's specific requirements in mind, it was also designed to be general enough to provide a suitable test bench for future, related projects. The general design objectives for VISILAB are presented in the following subsection.

6.1.1 VISILAB Design Objectives

VISILAB was designed to fulfil the following objectives [145], [146]:

- **Primary Objective:** To test the LION vision-based absolute navigation algorithm with camera hardware in-the-loop, placed over a lunar-like terrain.
- **Secondary Objective:** To provide ESA with an internal test facility for optical landing sensors, e.g. visible cameras or LIDAR systems, to enable the following activities to be carried out:
  - To allow for a comprehensive understanding of end-to-end error contributors to vision-based navigation systems, including offline and online processes, for pinpoint and soft landing applications.
  - Hardware-in-the-loop prototyping and testing of vision-based navigation and hazard detection systems, allowing the achievement of technology readiness level 3 according to Figure 6.1.
  - The characterisation of vision-based camera sensors and the impact of different detector technologies (e.g. rolling or global shutter) on image processing performance.

This current project was the first project to make use of VISILAB since its original inception and use in the LION project. Thus it is the first project to make direct use of the secondary design objective. However, due to a recent lab move, resulting in the loss of one of the two optical tables that VISILAB was originally installed upon, some changes had to be made to the configuration of the equipment before it could be used in this project. These changes (which will be detailed in Section 6.2) were made with the view to ensure continued fulfilment of the secondary objective.

6.1.2 Hardware Requirements

A set of hardware requirements were drawn up as part of the work carried out during the LION project in order to fulfil the design objectives presented in the previous subsection. These requirements are summarised in Table 6.1. For further details on the justification of the requirements see [145]. These requirements were used to select appropriate hardware for the VISILAB system.

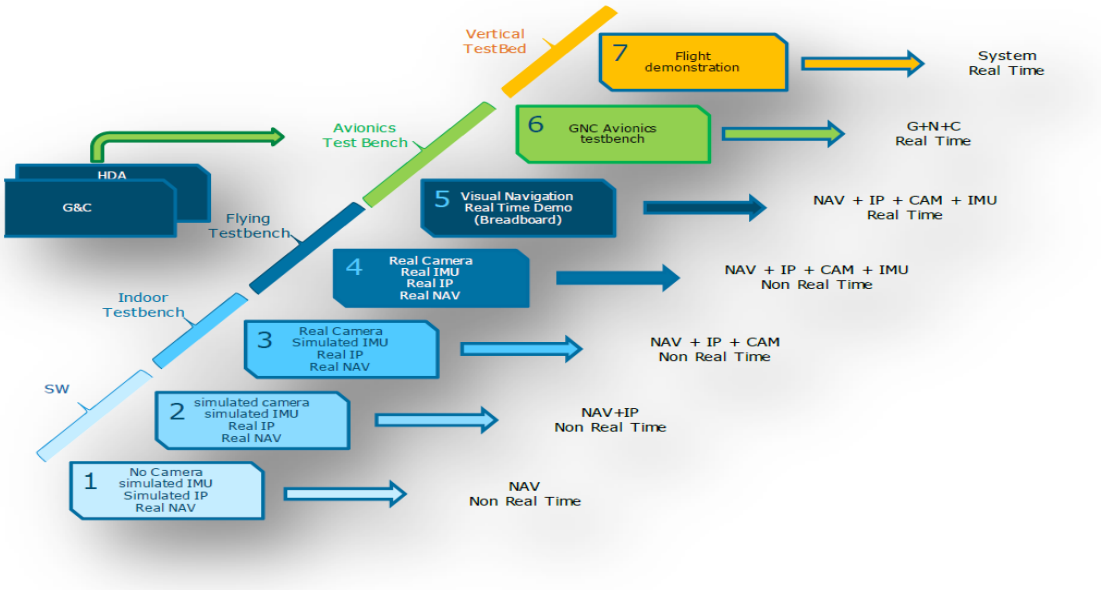


Figure 6.1: Technology readiness levels for vision-based navigation systems

Hardware	**Requirements**
Camera	1024x1024 pixel sensor 70 degree field-of-view
Inertial Measurement Unit	Software simulated
Planetary Surface Model	Lunar DEM from NASA LRO mission Real data only
Motion Capability	4 degrees of freedom: 3 translations + pitch rotation
Illumination	White light, parallel rays, uniform flux

Table 6.1: VISILAB hardware requirements.

6.1.3 Camera System

The camera system that was selected for VISILAB is a uEye USB camera manufactured by IDS Imaging, which was chosen to representative of the current industrial baseline for the ESA lunar lander camera [145]. It offers a USB 2.0 interface, a global-shutter mode, a 1280x1024 monochrome CMOS sensor, and is compatible with C-mount lenses. This camera was used with a 3.5mm lens, made by GOYO Optical Inc., for the LION experiment, although a number of other lenses are also available (however, the GOYO lens was also used in this project, so this is the only lens that will be discussed (see [145] for details on these other lenses)). The main parameters for the uEye camera with the GOYO lens are summarised in Table 6.2. The uEye camera, fitted with the GOYO lens is shown in Figure 6.2.

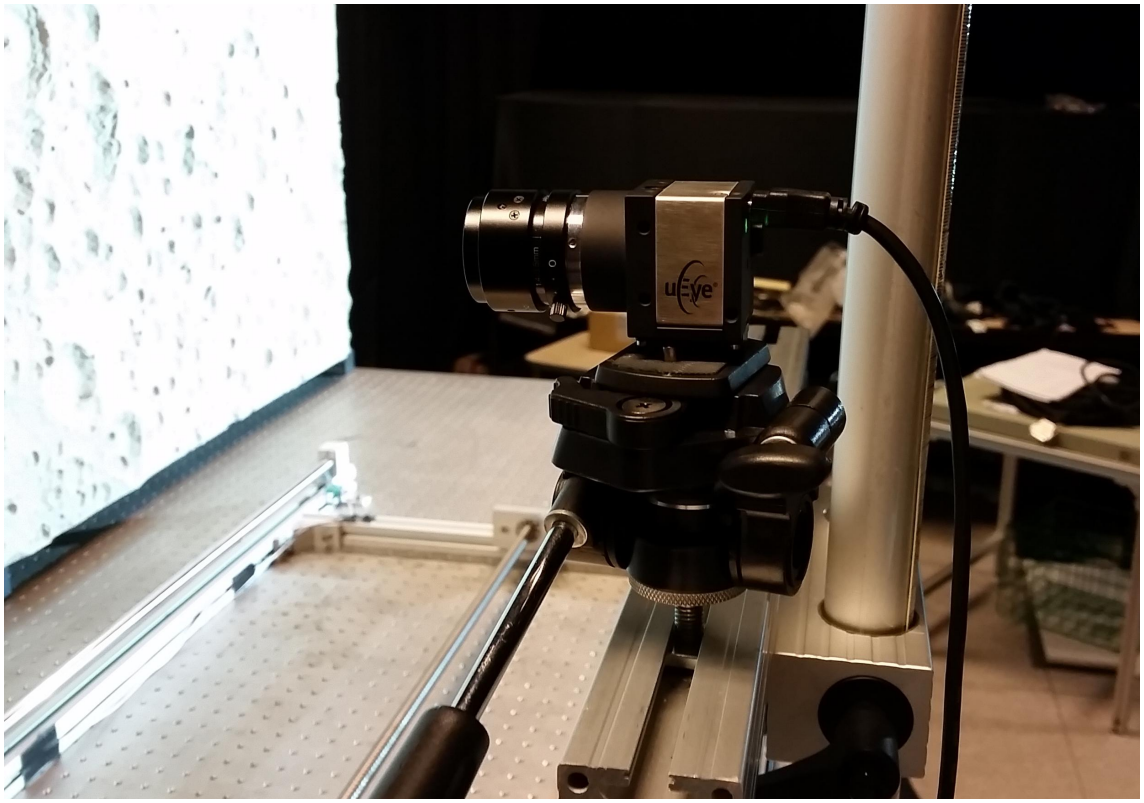


Figure 6.2: uEye camera fitted with GOYO lens

6.1.4 Planetary Surface Mock-Up

Since the LION project focussed on a descent navigation scenario for a pin-point landing on the Moon, the chosen terrain mock-up was that of a scaled portion of the Lunar south pole. The surface mock-up was produced for ESA by a specialised company, via the German Aerospace Center (DLR), that has developed techniques for milling precise surface mock-ups from real Digital Elevation Model (DEM) data. The DEM used to construct the VISILAB surface mock-up was created from altimetry data collected by the NASA Lunar Reconnaissance Orbiter mission (LRO) LOLA instrument. This DEM is shown in Figure 6.3, where the red box approximately indicates the region covered by

Camera Model	IDS UI-1240SE-M
Interface	USB 2.0
Resolution	1280x1024
Resolution Depth	8 bits
Sensor Size	6.784x5.427mm
Field of View	78.4x75.6
Minimum Distance of Focus	6cm

Table 6.2: Main camera parameters for uEye camera with GOYO lens

Exact Mock-Up Dimensions	980x1960mm
Milling Line Step	0.5mm
Maximum Height Range	50mm
DEM Resolution	1960x3920 pixels
DEM Resolution Depth	16 bits
Lunar Dimensions	960x1920km
DEM Lunar Pixel Footprint	490m

Table 6.3: Exact mock-up, DEM, and Lunar surface characteristics

the VISILAB mock-up surface, shown in Figure 6.4.

The mock-up surface consists of two 1m^2 tiles of reinforced resin. As can be seen by comparing Figures 6.3 and 6.4, the Lunar south pole is located in the left tile of the mock-up — in fact during the design process it was constrained to be exactly in the centre of the left tile. Each tile is milled with a milling strip resolution of 0.5mm and a depth range of 50mm. The footprint of the mock-up surface on the Lunar surface represents an area of 960x1920km. Scaling the raw DEM data down to a size of 1x2m resulted in a vertical height range of only 1.5cm. Thus, to take advantage of the full height range offered by the milling process, the DEM data was exaggerated by a vertical scaling factor to give the full 5cm range on the mock-up. The exaggerated DEM used to manufacture the mock-up is provided as a TIFF image (see Figure 6.5), where the grey value of each pixel represents the relative height of the terrain. This can be used to create a 3D terrain model using PANGU which may be useful in producing ground truth surface shape measurements for algorithms that aim to re-construct the surface shape from descent imagery.

The exact characteristics of the VISILAB terrain mock-up, DEM, and the Lunar surface region upon which it is based are summarised in Table 6.3.

6.1.5 Motorised Linear Table and Camera Mounting

The camera is mounted on a tripod head, which is in turn mounted onto a vertical support on top of the linear table as shown in Figure 6.6. The whole mock-up and linear table arrangement is mounted on an optical table, which allows for accurate measurements of the set-up of the equipment to be made.

The tripod head allows for adjustment of the pitch angle, as indicated by the green arrow in Figure 6.6. Adjustment of the roll or yaw is also possible by using the handle on the tripod head to adjust the tilt of the camera (depending on the orientation of the tripod head with respect to the vertical support – in Figure 6.6 the handle would adjust the yaw angle, as indicated by the red arrow) However, adjustment of these angles is difficult to achieve with any reliable precision or accuracy, especially if many continuous small adjustments need to be made during the course of the trajectory. To do this, one would need to make the adjustment by aligning the camera manually to the required angle by eye, using a protractor, which would lead to significant uncertainty in the angle measurement. It is possible to set an initial angle, and use images of the calibration pattern (set in a known location on the optical table) to calculate the initial orientation of the camera with respect to the terrain (see Subsection 6.4.3 for details). However, during

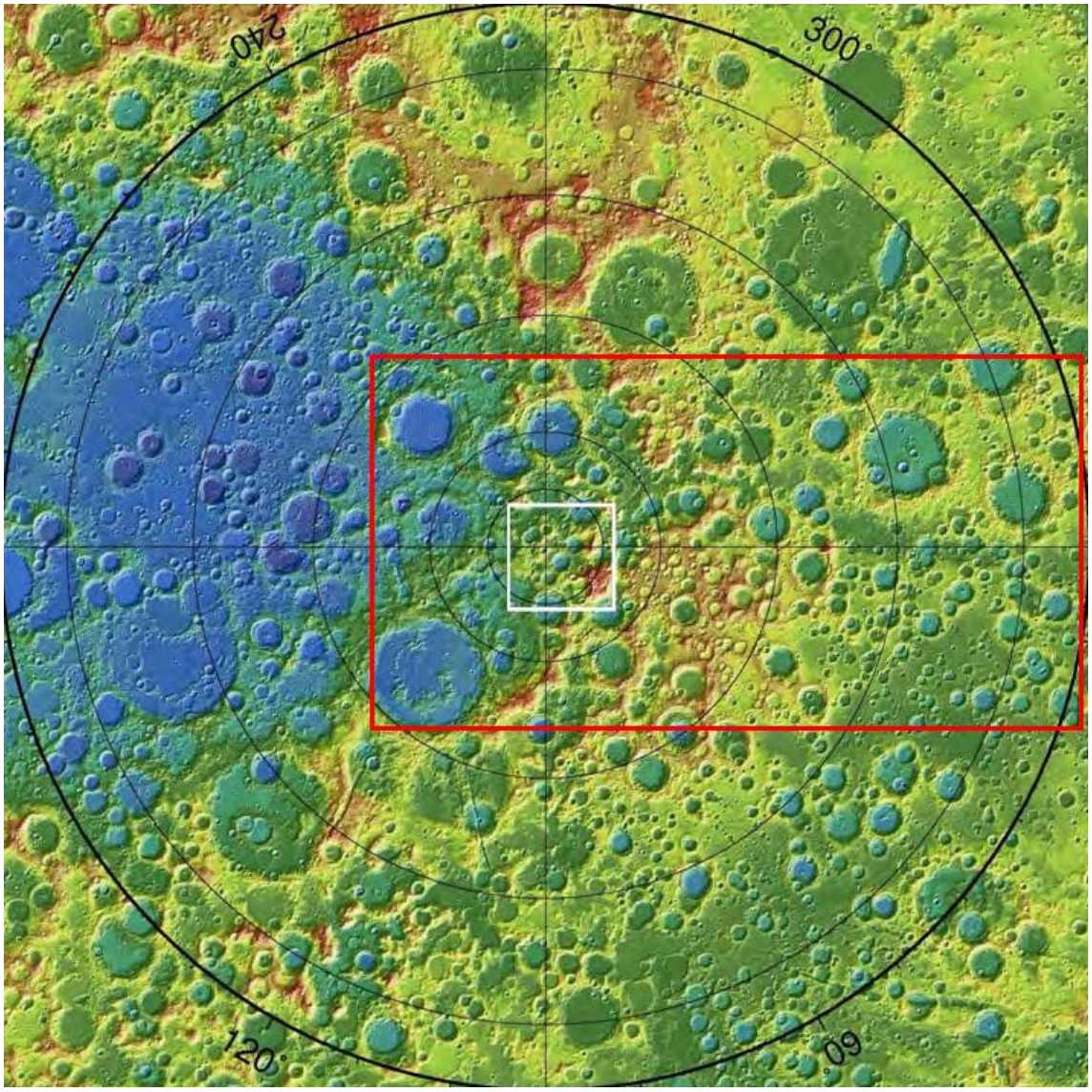


Figure 6.3: LRO LOLA DEM of Lunar south pole — red box indicates VISILAB mock-up terrain

the recording of the trajectory, the calibration pattern must either be removed so that it does not obscure the view of the terrain, or if this is not an issue (i.e. the calibration pattern is in its normal location, attached to the side of the mock-up), then the movement of the camera will quickly result in the calibration pattern being outside of the field-of-view of the camera. Therefore, regular and repeated calculation of the orientation of the camera during the course of the trajectory would be an extremely tedious and time consuming process (requiring movement of the calibration pattern to inside the field-of-view each time the camera orientation is changed, recording a number of images of the pattern, calculating the orientation, followed by removal of the pattern again in order to proceed with the trajectory), to the point where it becomes impractical.

In terms of translational motion, there are a further 3 degrees of freedom. The camera

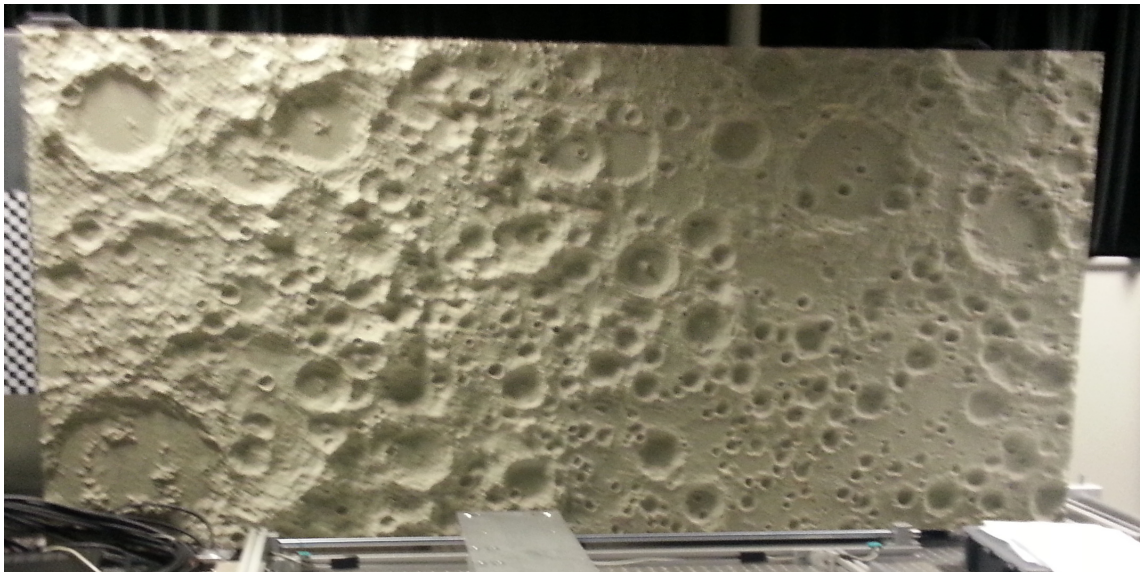


Figure 6.4: VISILAB surface mock-up – The terrain mock-up actually consists of two separate 1m^2 tiles, tightly fastened side-by-side

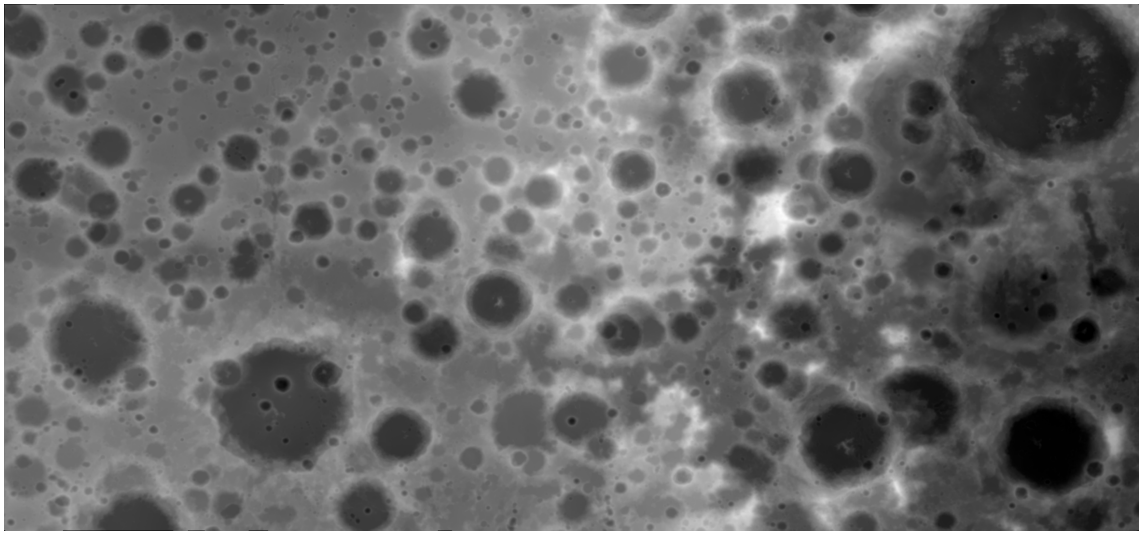


Figure 6.5: TIFF image that can be used to construct a ground truth DEM of the VISILAB terrain. NOTE: This is rotated by 180 degrees with respect to the physical mock-up.

height can be adjusted to provide cross range motion – adjustments can be measured using the tape measure that is fixed along the vertical support. The 2-axis linear table provides the remaining two degrees of freedom through the use of two motorised lead screws. These are arranged perpendicular to each other to provide translational motion in the x- and y-directions, as indicated in Figure 6.6, which correspond to downrange motion and altitude, respectively. Tape measures have also been attached to the runners for these two directions to enable measurements to be recorded, however measurements are also provided by the motor control software.

6.1.6 Reference Frames

There are four reference frames to consider when working with VISILAB, each of which is described in this subsection and shown in Figure 6.7 and 6.8.

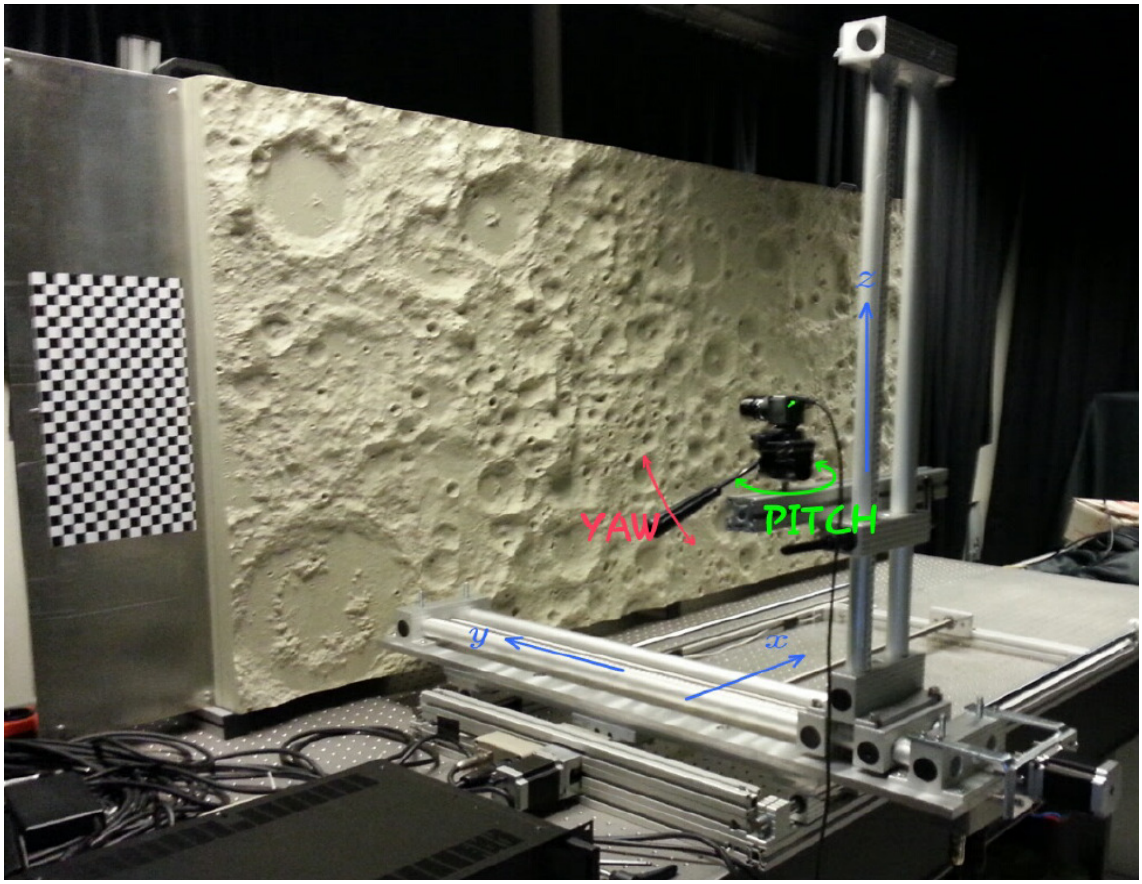


Figure 6.6: Linear table and camera mounting structure. The way in which the pitch and yaw angles can be adjusted, and the x, y, and z directions of motion of the camera are also indicated.

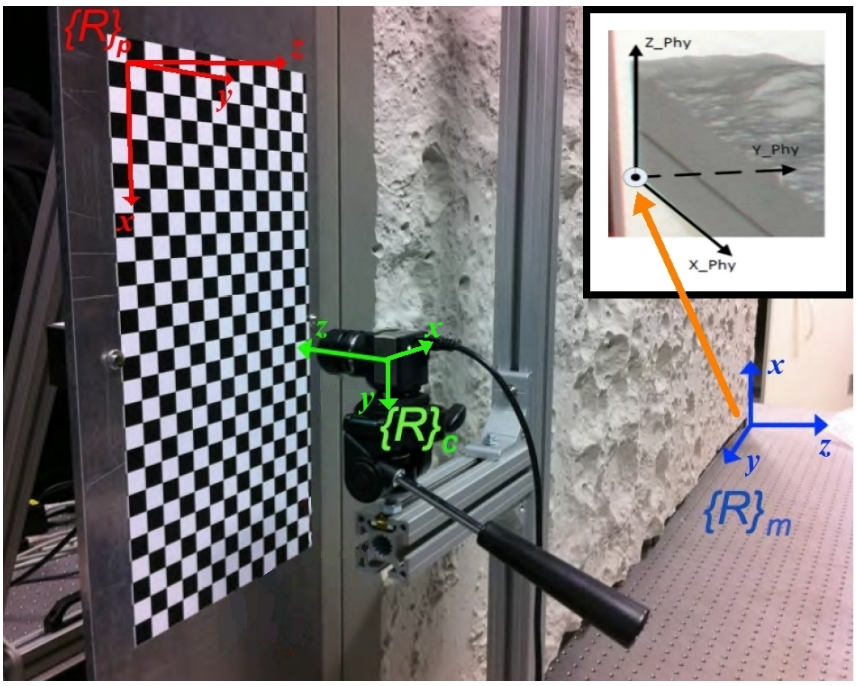


Figure 6.7: VISILAB reference frames – mock-up frame $\{R\}_m$, calibration pattern frame $\{R\}_p$, camera frame $\{R\}_c$ – Inset: The mock-up frame coordinate system origin is actually located on the back of the tile

Mock-Up Frame $\{R\}_m$

- *Origin*: Bottom right corner of the mock-up when facing it, on the back side.
- *X-direction*: Vertical upwards.
- *Y-direction*: Horizontal leftwards.
- *Z-direction*: Horizontal towards front of mock-up.

This coordinate system was defined during the construction of the terrain surface. The purpose of this frame is to enable the camera pose to be expressed relative to the terrain surface (see [147]). The reason why the origin is on the back side of the tile is because this provides a flat surface from which to take measurements. However, since a different method will be described in this report to record the camera pose, the mock-up frame's significance is now somewhat lessened. Its main purpose now is in its relation to the coordinate frame used in the TIFF image DEM. The reference frame of the TIFF DEM is shown in Figure 6.9 and the differences between this and the mock-up frame are summarised in Figure 6.10 (see [147] for further details).

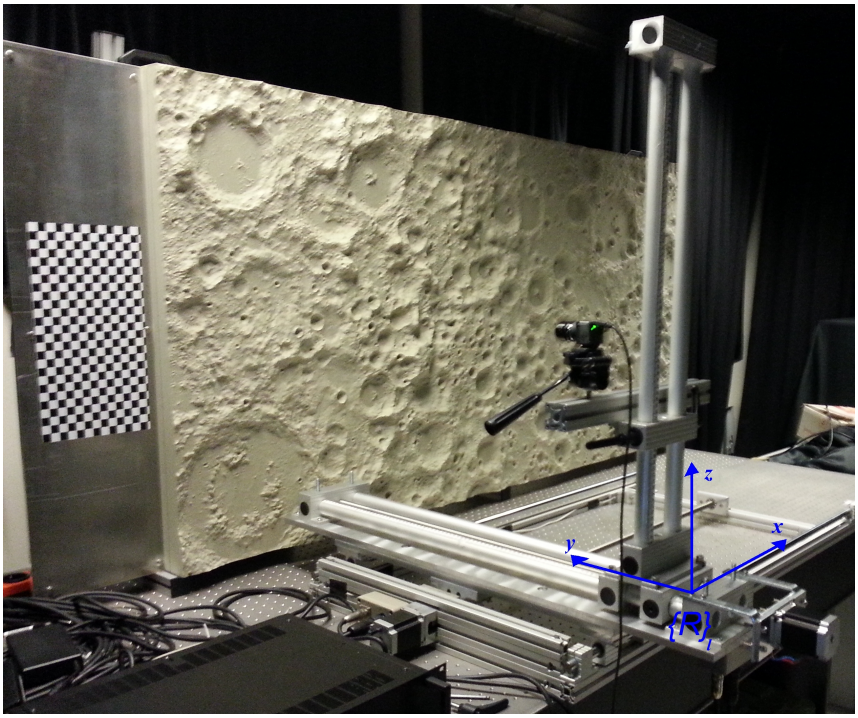


Figure 6.8: VISILAB reference frames – linear table frame $\{R\}_l$

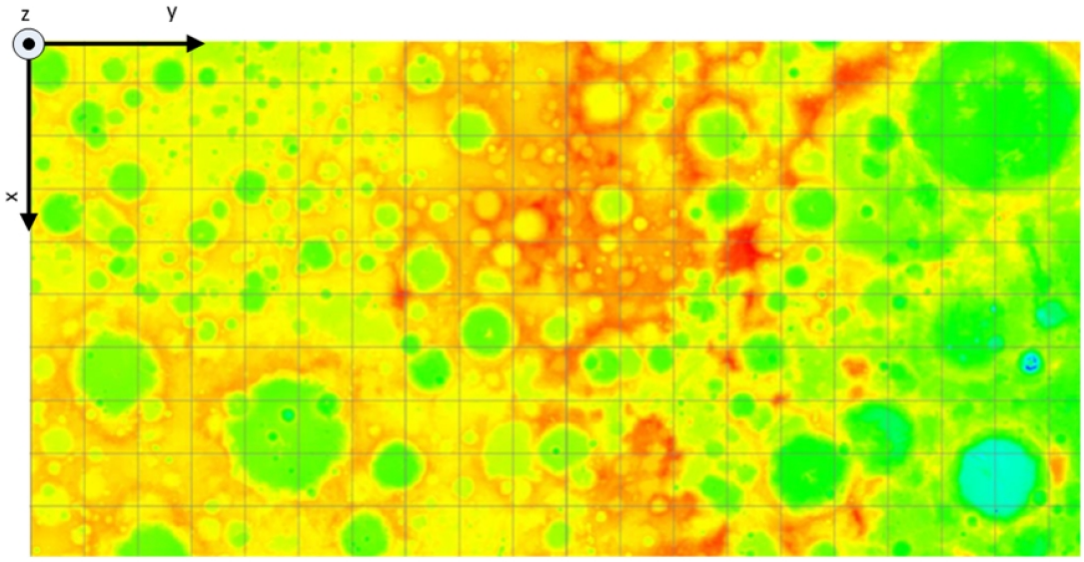


Figure 6.9: TIFF DEM Reference frame

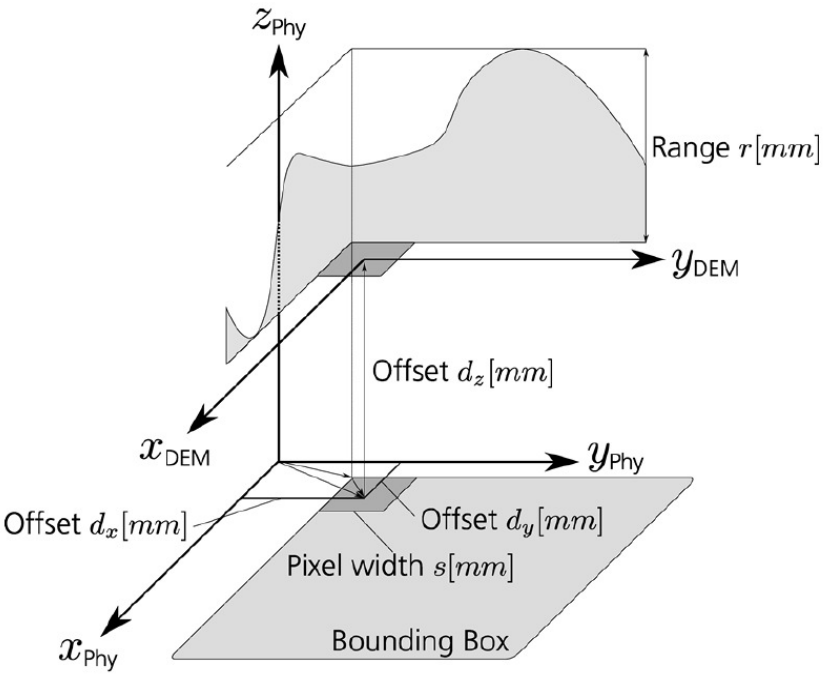


Figure 6.10: Comparison of TIFF DEM and physical mock-up reference frames

The parameters in Figure 6.10 are as follows:

- The width of a pixel, s, in the DEM represents 0.500mm on the physical mock-up.
- The height range, r, in the DEM corresponds to a physical range of 44.730mm on the mock-up.
- The x and y offsets, d_x and d_y, of the centre of the (0, 0)th pixel is 0.247mm and 0.232mm, respectively. This offset is composed of the data bounding box with respect to the physical reference frame and half a pixel width, in each direction.
- The z offset, d_z, between the lowest point in the TIFF DEM and the backplane of the physical mock-up is 54.938mm.

The light grey square in Figure 6.10 represents the data bounding box (i.e. the whole TIFF image) in physical reference coordinates and the dark grey square represents the (0, 0)th pixel in the image (with its projection onto the x_{Phy}, y_{Phy} plane, to better illustrate the relationship between the two reference frames and the components that describe the differences).

This relationship between the two coordinate systems results in the following equations for calculating the physical 3D coordinates of a point on the mock-up from its corresponding point in the TIFF image [147]:

$$\begin{aligned} x_{Phy} &= d_x + si \\ y_{Phy} &= d_y + sj \\ z_{Phy} &= d_z + r\frac{I(i,j)}{2^{16}} \end{aligned} \tag{6.1}$$

where i and j denote the $(i,j)^{th}$ pixel and $I(i,j)$ is the grey value of the $(i,j)^{th}$ pixel in the TIFF image.

Calibration Pattern Frame $\{R\}_p$

VISILAB has a built-in calibration pattern for calibrating the camera. The calibration pattern is adhered to an aluminium plate, which is in turn mounted on the terrain support structure to the left of the terrain (when facing the mock-up). This reference frame is positioned and oriented as shown in Figure 6.7, i.e.:

- *Origin*: 1 square in from the top left corner of the calibration pattern (although this depends on the first corner clicked during camera calibration — see Section 6.3).
- *X-direction*: Vertical downwards.
- *Y-direction*: Horizontal rightwards.
- *Z-direction*: Horizontal towards the front of the mock-up.

The calibration pattern frame is an important reference frame in the use of VISILAB as it is used in the determination of the initial position of the camera with-respect-to the mock-up surface. This is discussed further in Subsection 6.4.3.

Camera Frame $\{R\}_c$

The camera reference frame is extremely important in the use of VISILAB, as it is in all computer vision related tasks. In general, all low-level measurements of the scene via the use of images (e.g. tracked feature points) are computed in the image coordinate system (similar to the reference frame for the TIFF DEM shown in Figure 6.9) by computer vision algorithms. The image coordinate system is related to the camera frame via the intrinsic parameters of the camera system. It is generally much more useful to express the measured quantities in the coordinates of the camera reference frame since this reference frame exists in normal 3D space, whereas the image coordinate frame exists in image space. The camera reference frame in VISILAB has the following properties:

- *Origin*: Optical centre of the camera.
- *X-direction*: Defined by the light sensing array of the camera. Approximately to the right when the camera is set up as shown in Figure 6.7.
- *Y-direction*: Also defined by the light sensing array of the camera — ideally this will be perpendicular to the X-direction. Approximately vertically downwards in Figure 6.7.
- *Z-direction*: Along the optical axis, towards the observed scene.

Linear Table Reference Frame $\{R\}_l$

This is the reference frame used in commanding the motion of the camera through the motor controller software. Due to the fact that all of the equipment is mounted on an optical table, it is assumed that the x-axis of the linear table is perfectly parallel but opposite to the y-axis of the mock-up frame, the y-axis of the linear table is perfectly parallel but opposite to the z-axis of the mock-up, and the z-axis of the linear table is perfectly parallel to the x-axis of the mock-up. Therefore the motion of the camera in the linear table frame is trivially related to the motion in the mock-up frame (due to this assumption, in the current project the mock-up frame is not used as a base for describing the motion of the camera — only the linear table frame is used). The properties of the linear table frame are summarised as follows:

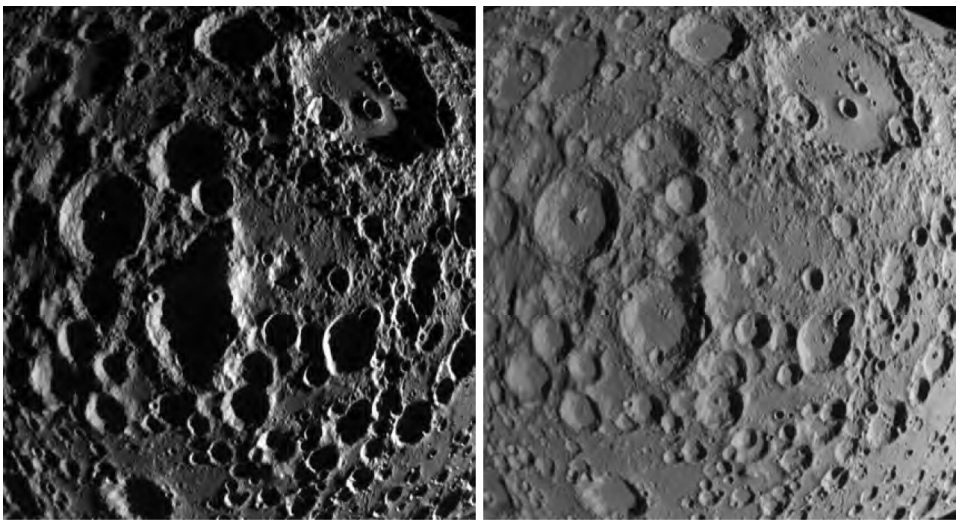
- *Origin*: Located at the home position of the linear table — i.e. when the camera is at its leftmost position and furthest away from the mock-up surface.
- *X-direction*: Horizontal to the right.
- *Y-direction*: Horizontal towards the mock-up.
- *Z-direction*: Vertically upwards.

6.1.7 Illumination System

The light source selected for VISILAB is a simple overhead projector. A number of other possibilities were considered in order to satisfy the requirement for parallel rays of white light with uniform flux, but these were ruled out due to impracticalities and cost constraints. The closest one can get to fulfilling the requirement when taking into account cost and space constraints is to make use of a cinema/theatre spot lamp, which makes use of a halogen lamp placed behind a Fresnel lens in order to achieve low divergence. This is the same principle used in overhead projectors, and since one was already available at ESTEC, this was the chosen solution.

It was found that using an overhead projector allowed for fairly realistic images to be recorded, such as those in Figure 6.11. However, a slight problem does arise with respect to flux variation across the surface, particularly when the light source is positioned such that the incidence angle is large (i.e. when the projector is over to one side of the mock-up in order to lengthen shadows). This phenomenon is easily explained when one considers the light source as a point source close to the surface. In this case, the illumination flux decreases as an inverse quadratic function of the distance. Another problem arises due to the proximity of the light source to the surface that is necessitated by the space constraints in the lab. Although the effects of non-parallel rays were reported not to be visible to the naked eye [145], the effects of this may manifest itself in algorithms that depend on the illumination direction. Because of the non-parallel nature of the rays arising from the point light source at a close distance, the illumination angle will vary across the surface. Both of these phenomena are not representative of the true lunar conditions and are thus a source of difficulty, especially in algorithms that depend strongly on the illumination conditions, such as shape-from-shading algorithms.

The light source is shown in operation in Figure 6.12.



(a) Sun elevation: 10 deg. (b) Sun elevation: 30 deg.

Figure 6.11: Example images of the same area in VISILAB with different illumination directions

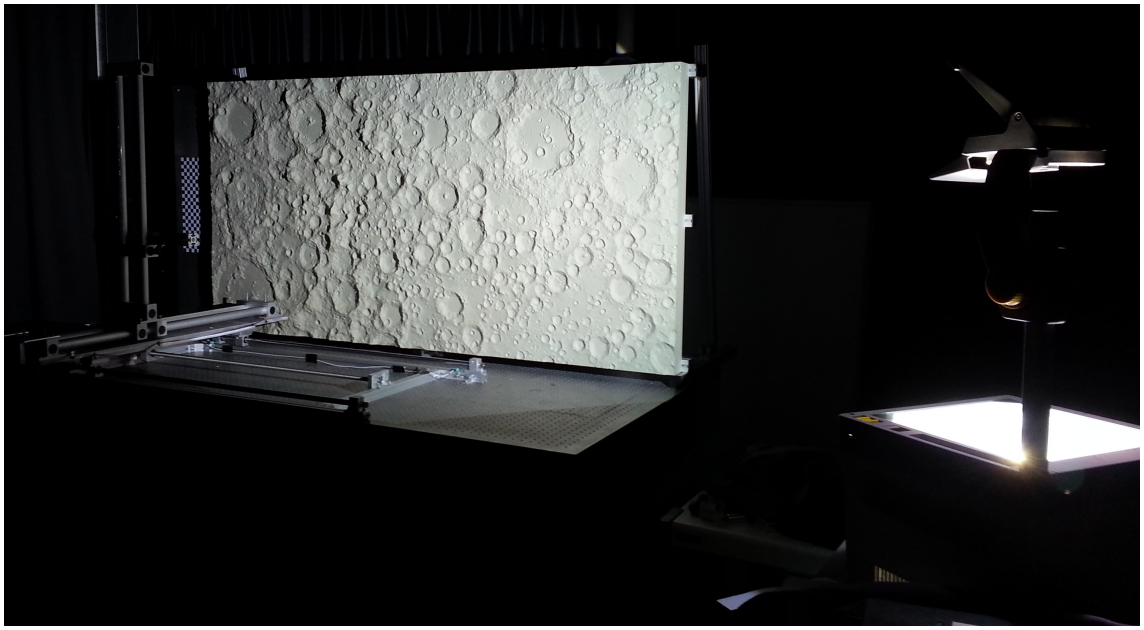


Figure 6.12: VISILAB with light source in operation

6.2 Reconfiguring VISILAB

Upon arrival at ESTEC, the VISILAB equipment had recently been moved from a previous location in another lab. In order to move the equipment it had to be partially disassembled, and by February 2014 it had not yet been reassembled in its new location. In its previous location, the apparatus was set-up spanning two optical tables, whereas in the new lab only one optical table was available. This chapter describes the configuration changes and decisions that had to be made in order for the equipment to be accommodated in its new location, whilst still fulfilling the design objectives as best as possible.

6.2.1 VISILAB Configuration Options

The VISILAB apparatus was originally set up on two optical tables, but due to a resent lab move it had been partially disassembled and moved to a location where only one optical table was available. Therefore, a new configuration was required to make best use of the space available. A number of CAD models were created to illustrate the potential set up options and how these configurations would impact on the space available in the lab with respect to other nearby objects. Figure 6.13 shows three views of the first configuration option. The left view shows the configuration from a top down perspective and illustrates the location of the VISILAB equipment with respect to the available free space in the lab (indicated by the 4 walls and a nearby computer cabinet (bottom right of the view)). This configuration provides the most flexibility in terms of the positioning of the illumination source (not shown) as it would allow for a large variation in illumination angle without having to place the light source too close to the terrain. It also

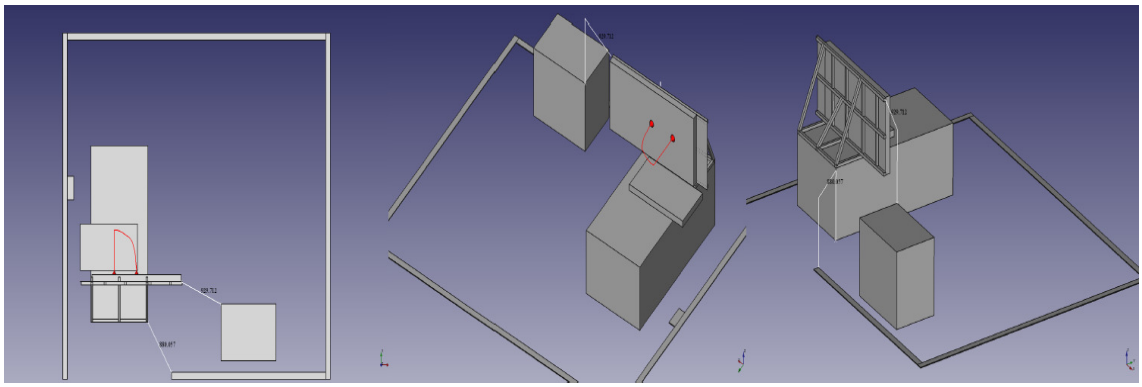


Figure 6.13: VISILAB configuration option 1.

provides a great deal of flexibility in the positioning of the linear table, since it leaves a large area of the optical table free of other equipment, therefore the camera can be positioned at a significant "height above" the terrain. However, this configuration requires that a significant portion of the terrain support structure overhangs the edge of the optical table. While this would most likely be acceptable in terms of the rigidity of the structure, it creates a relatively small gap between the edge of the terrain and the nearby computer cabinet (~ 0.93m), which may cause issues with accessibility to the storage area behind the equipment (the entrance to this storage area is illustrated by the gap in the wall). Having said this, the gap between the optical table and the entrance to the storage area is only ~ 0.88m, therefore this configuration does not impede access any further than the optical table does by itself. However, having such a large overhang of the terrain structure may be objectionable on health and safety grounds, since numerous objects are stored in boxes on the floor underneath the optical table (different camera lenses, VISILAB spare parts, and tools, etc.), which would frequently create the need for a person to temporarily position themselves underneath the overhang. Additionally, having the terrain overhanging the optical table, the risk of damage to the quite fragile, and precision engineered terrain surface is also unacceptably high.

Figure 6.14 shows another potential arrangement of the equipment. In this case the

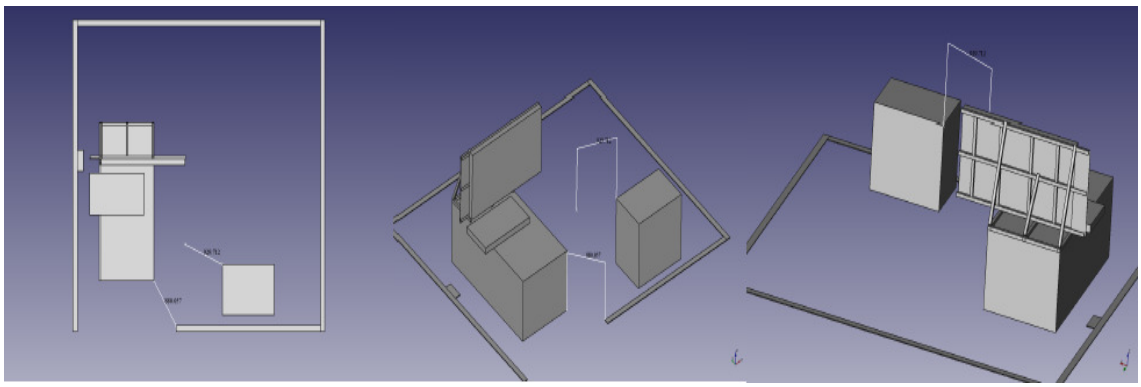


Figure 6.14: VISILAB configuration option 2.

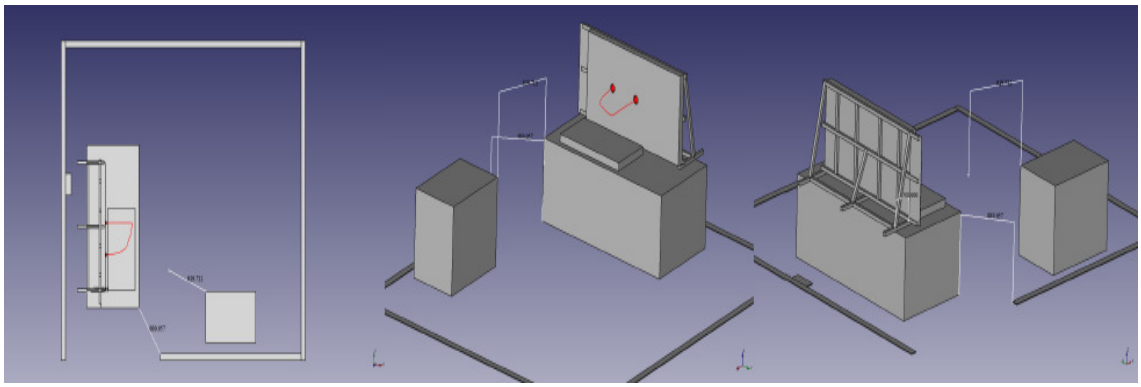


Figure 6.15: VISILAB configuration option 3.

terrain support structure is placed at the opposite end of the optical table. While this eliminates the issues with accessibility to the storage area, this configuration has the least amount of flexibility in terms of positioning the light source due to the proximity of the computer cabinet and storage area. Another issue with this configuration and the previous one is that it requires permanent modification of the support structure, which is undesirable. The potential health and safety issue of the previous configuration is also present in this configuration. The risk of damage to the terrain surface, while reduced due to it no longer obstructing a thoroughfare, is also present in this configuration.

Figure 6.15 shows the third identified configuration. Whilst the flexibility of the positioning of the motorised linear table (illustrated by the box in front of the terrain) is greatly reduced, leading to a potentially restrictive limit on the available "height" range of the camera, this arrangement, unlike the other configurations, requires no permanent modifications to the terrain support structure. It also still allows for a large degree of flexibility in the light source position. Primarily due no significant changes being required to the terrain support structure, this option was chosen for the new configuration of the VISILAB test bench.

6.3 Camera Calibration

In almost all computer vision problems, the camera must be pre-calibrated in order to obtain metric measurements of objects in the scene. This is an extremely important part of tackling the problem, since the accuracy of the calibration affects the accuracy of the results. Camera calibration is usually the first practical step in tackling a computer vision problem. Based on the theory and reasoning behind camera calibration presented in Chapter 2, this section outlines the procedure for calibrating the camera in VISILAB using a software tool known as the "Camera Calibration Toolbox for Matlab", followed by the results of the calibration.

The purpose of camera calibration is primarily to estimate the intrinsics matrix. This is achieved by taking multiple images of a calibration pattern, such as those shown in Figure 6.16, which consist of a black and white chequerboard pattern with known square size. Since the square size is known, these images can be used to solve for the intrinsics

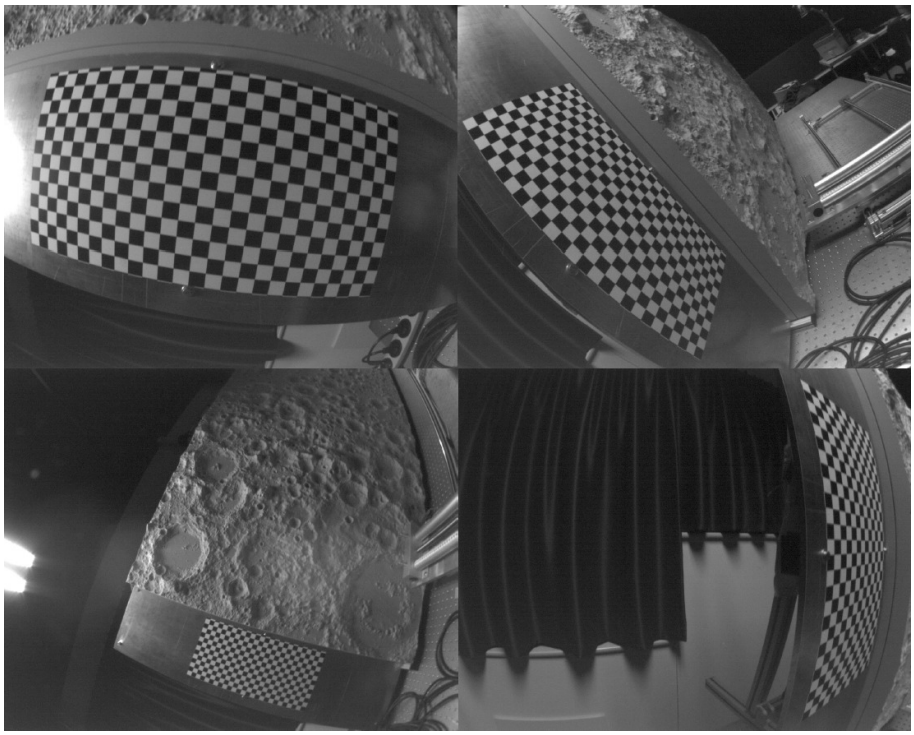


Figure 6.16: Example calibration images — also shows presence of image distortion

matrix. Due to the way in which the problem of camera calibrations is formulated, the extrinsic parameters of are also computed for each image, though these are usually not important in most cases. Additionally, if significant lens distortion is present in the images (as is the case in Figure 6.16), the calibration method may also be used to simultaneously solve for the lens distortion coefficients. This would then allow for the images to be undistorted, as shown in Figure 6.17, which then effectively allows the pin-hole camera model to be applied in any subsequent computer vision algorithms.

6.3.1 VISILAB Camera Calibration Procedure

As discussed previously, the VISILAB apparatus includes a camera calibration pattern to the left of the mock-up. This consists of a chequerboard pattern of 12x24 (usable) black and white squares of size 15x15mm. The calibration process involves recording multiple images (e.g. 30) of the calibration pattern from various distances and angles, which can be achieved by removing the camera from the translation table apparatus and positioning it somewhat randomly by hand for each image. It is important that the calibration pattern is entirely within the field-of-view of the camera for each image and that a wide range of angles and distances are used so that over the calibration image sequence as a whole the calibration pattern has been viewed in positions covering the whole field of view of the camera. The procedure for acquiring calibration images is summarised as follows:

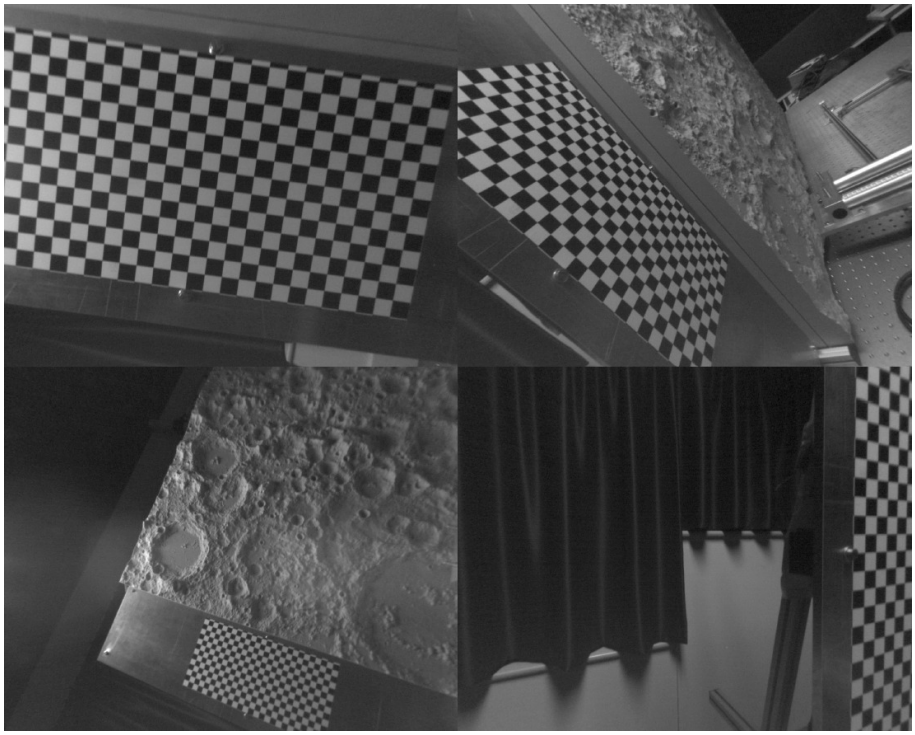


Figure 6.17: Example calibration images after undistortion — Note: The images have been cropped slightly to remove the borders that arise from the rectification process

Calibration Image Acquisition Procedure

1. Remove camera from vertical support of translation table apparatus
2. Hold camera at a certain position and orientation such that calibration pattern is entirely within the field-of-view
3. Record image using uEye software.
4. Repeat steps 2 and 3 until sufficient images have been recorded, ensuring that over the whole sequence the calibration pattern will be viewed from positions and angles with roughly equal coverage of the entire field-of-view and of the entire working distance range.
5. Replace the camera back on the vertical support in the starting position and orientation required for trajectory data capture and record several more images (e.g. 10) of the calibration pattern (providing it is entirely within the field-of-view of the camera) with the camera in this initial position – The importance of these images will be discussed in the following Chapter. These images may be used in calculating the camera calibration parameters or they may be kept separate and used only for their intended purpose.

The image processing and computation of the intrinsic and extrinsic camera parameters from the calibration images can be carried out using the Camera Calibration

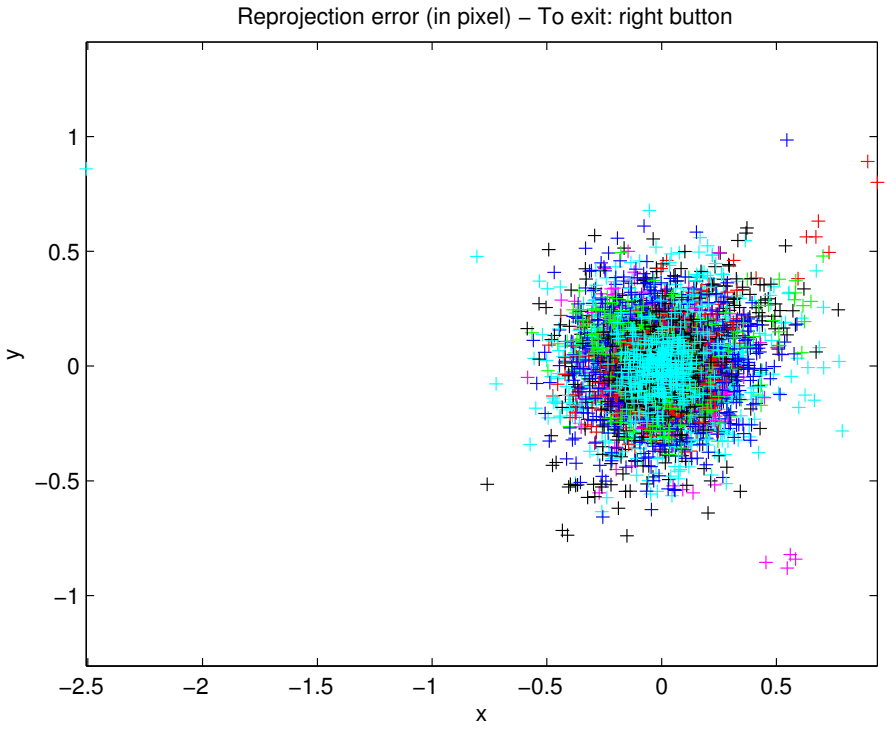


Figure 6.18: Corner re-projection errors

Toolbox for Matlab developed by Jean-Yves Bouguet. Full details on how to use this toolbox, and especially for details on the additional steps required when dealing with highly distorted images, can be found at the Camera Calibration Toolbox website (http://www.vision.caltech.edu/bouguetj/calib_doc) in the calibration examples section (first calibration example), thus these steps will not be repeated here.

The calibration procedure described on the Camera Calibration Toolbox website provides a number of tips for refining the camera calibration results after the initial calibration. These extra steps should be repeated a number of times until a satisfactory pixel error is obtained, and the number of outliers are seen to be very few when observed using the 'Analyse Errors' visualisation tool provided in the toolbox and shown in Figure 6.18.

The results obtained from the calibration procedure with an acceptable level of accuracy (although improvements may still be possible) are shown in Table 6.4. By examining the errors in the calibration results Table 6.4 shows that a very reasonable calibration accuracy has been achieved. This can be further seen in Figure 6.18, which presents the re-projection errors obtained by using the calibration parameters to re-project the corner points of the calibration pattern onto the each of the images. The re-projection errors in Figure 6.18 can be seen to be very densely clustered between ± 0.5 pixels and centred on zero, in both the x- and y-axes, with a roughly circular distribution, indicating that the errors are approximately zero mean and uncorrelated, i.e. the errors can be approximately characterised by a standard Gaussian distribution. A

Camera Parameters	**Estimated Value with 3σ Errors** (units = pixels)
Focal Length	$\begin{bmatrix}691.30506 & 691.53036\end{bmatrix} \pm \begin{bmatrix}0.45377 & 0.45117\end{bmatrix}$
Principal Point	$\begin{bmatrix}619.21494 & 532.83860\end{bmatrix} \pm \begin{bmatrix}0.44702 & 0.42917\end{bmatrix}$
Skew (units = degrees)	$\begin{bmatrix}-0.00031\end{bmatrix} \pm \begin{bmatrix}0.00011\end{bmatrix} \implies$ angle of pixel axes $= 90.01804 \pm 0.00629°$
Distortion	$\begin{bmatrix}-0.34448 & 0.14584 & -0.00052 & -0.00017 & -0.03189\end{bmatrix} \pm \begin{bmatrix}0.00074 & 0.00113 & 0.00008 & 0.00008 & 0.00050\end{bmatrix}$
Pixel Error	$\begin{bmatrix}0.17526 & 0.16527\end{bmatrix}$

Table 6.4: Calibration results

small number of outliers can be seen surrounding the main cluster with a maximum error of approximately 2 pixels. This plot helps to visualise the pixel error given in Table 6.4, which is the 3σ error on the x- and y-axes, therefore 99.7% of all the calibration pattern corners were re-projected within $(0.17526, 0.16527)$ of the mean, which is approximately $(0, 0)$. This again confirms that an accurate calibration has been achieved.

One final point that is worth mentioning relates to how the origin of the calibration pattern coordinate frame is set during the calibration process. One of the first steps in processing the calibration images involves selecting the outermost corners of the calibration pattern. The first corner clicked is used as the origin of the calibration pattern coordinate system. In this project the first corner clicked was the top left corner, which can be seen in Figure 6.7. You are free to choose any of the other 3 corners, however you should be consistent across all images. This was especially important in this work because the calibration pattern frame is used as the origin for the entire mock-up instead of using the previously mentioned mock-up frame.

6.4 Trajectory Computation

The purpose of VISILAB is to enable the development and testing of computer vision algorithms to aid in navigation and hazard detection during entry, descent and landing operations. Therefore, to correctly test the potential of such algorithms, the motion of the camera must be representative of a realistic descent trajectory in order to replicate as closely as possible the expected motion characteristics of the spacecraft in its intended working environment. This section discusses how the representative trajectories were computed, and then describes how this can be applied to VISILAB in order to obtain an image dataset.

6.4.1 Trajectory Design Considerations

Although the VISILAB mock-up surface is based on real altimetry data collected over the south pole of the moon and constructed at a specific scale, the use of VISILAB need not be restricted to Lunar south pole descent and landing scenarios. The terrain mock-up

may instead be viewed as any generic representative planetary surface, with a different selection of scale factor based on the specific requirements of the mission scenario that is being modelled. The overriding goal in the design of VISILAB was to provide a representative planetary surface, which, coupled with the light source, enables realistic images to be recorded using actual camera hardware. Therefore, the most important thing to consider when using VISILAB is the determination of the scale factor, since this affects all aspects of the motion and position of the camera when tracing out the calculated trajectory. Another aspect that needs to be considered is the camera frame rate - i.e. the number of images per second. The exact position of the camera relative to the terrain needs to be known at the point that each image is acquired. Not only should the camera frame rate be representative of what can be realistically achieved given the amount of image processing that is required by the algorithm, but also, the camera frame rate, coupled with the scale factor, determines the magnitude of the inter-frame displacement of the camera. If the frame rate is too high the inter-frame motion will be small and there is a limit to the level of precision that can be achieved in positioning the camera using the linear table motor controller software.

6.4.2 Trajectory Calculation

This subsection describes the calculation of a representative trajectory for subsequent application to the VISILAB test bench. To begin with, this trajectory is computed at full scale and then afterwards it will be scaled down to the VISILAB dimensions. Although VISILAB may be used to simulate a landing on any planetary body, as mentioned above, in this particular case we will in fact be considering a Lunar landing, but not necessarily at the Lunar south pole. Thus, we will be treating the terrain mock-up as a generic representative portion of the Lunar surface and setting our own scale factor according to the unique characteristics of the considered scenario. The trajectory described in this subsection was based upon the type of trajectory being considered in the ALHAT project [9, 10, 148, 149], which itself shares many similarities with the Apollo Moon landings. Therefore, the spacecraft upon which the calculations are based is the Apollo Lunar Module, which was chosen due to readily available data.

Figure 6.19 illustrates the key features of the trajectory, by outlining the different stages of the descent. The entire trajectory is one continuous powered descent, that has been split into five separate stages based on the pitch program of the spacecraft during certain periods of time. The first stage is a gravity turn manoeuvre starting from a circular parking orbit and continuing until a certain point above an altitude of 2000m. This is followed by a constant rate pitch-up manoeuvre lasting approximately 10 seconds such that the pitch angle is 100 degrees at 2000m altitude. This pitch angle provides a 35 degree look angle (see Figure 6.20), ensuring that the landing site is within the field-of-view of the camera and/or providing direct line-of-sight to the landing site for any crew on-board. It is at this point that hazard detection begins. Following this pitch-up manoeuvre, the spacecraft maintains this pitch angle until it reaches an altitude slightly above 50m, at which point another pitch-up manoeuvre begins to reduce the pitch angle to 90 degrees. At the conclusion of this second pitch-up manoeuvre, the spacecraft will have come to rest at an altitude of 50m, directly above the pre-specified landing site.

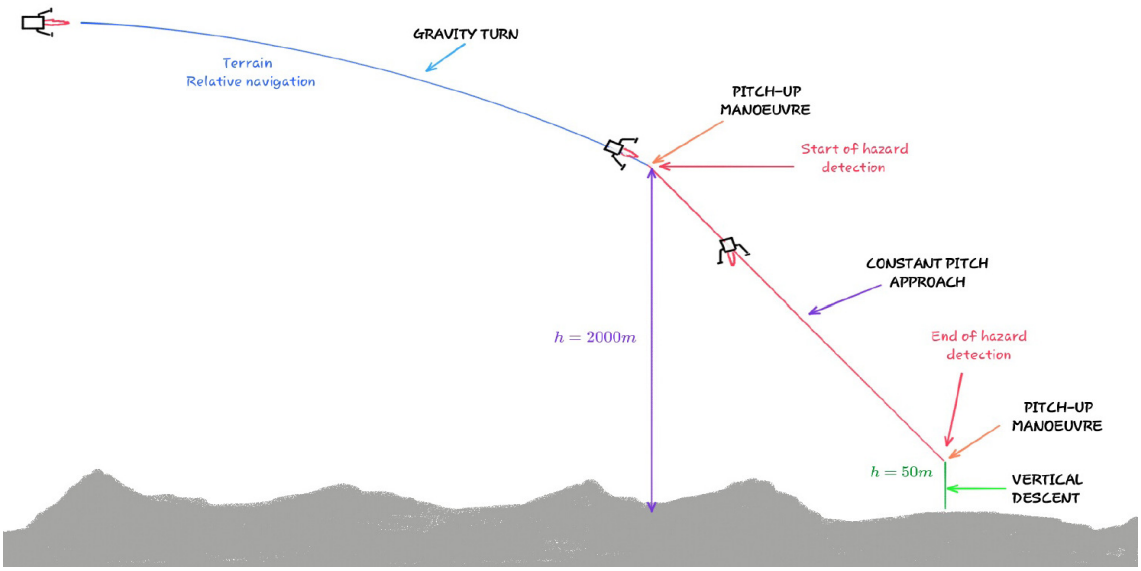


Figure 6.19: Lunar descent trajectory scenario

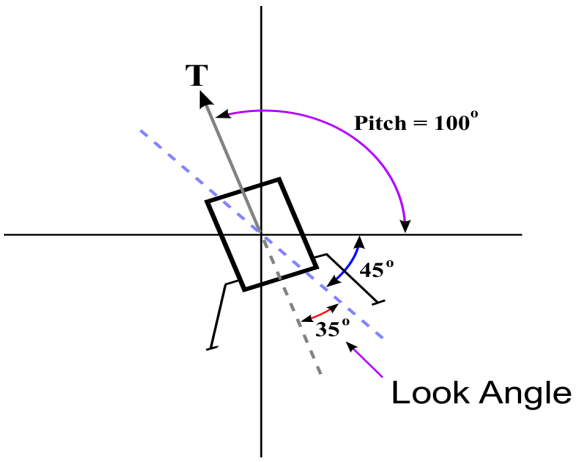


Figure 6.20: Look angle after first pitch-up manoeuvre

This is the point at which hazard detection must be complete in order to allow for the possibility of any required hazard avoidance manoeuvres to be carried out, as illustrated in Figure 6.21. Following this, the spacecraft will descend at a constant rate of 1m/s to a safe touchdown as close as possible to the pre-selected landing site. The part of the trajectory that was important in this project is the point from 2000m altitude down to the beginning of the vertical descent phase, since this is where hazard detection occurs. However, all phases of the trajectory are important and need to be calculated since this enables the motion during the hazard detection phase to be correctly characterised.

The trajectory is calculated by numerically integrating the following equations and applying constraints to either the pitch or altitude or both as the spacecraft transitions between the different phases of the descent:

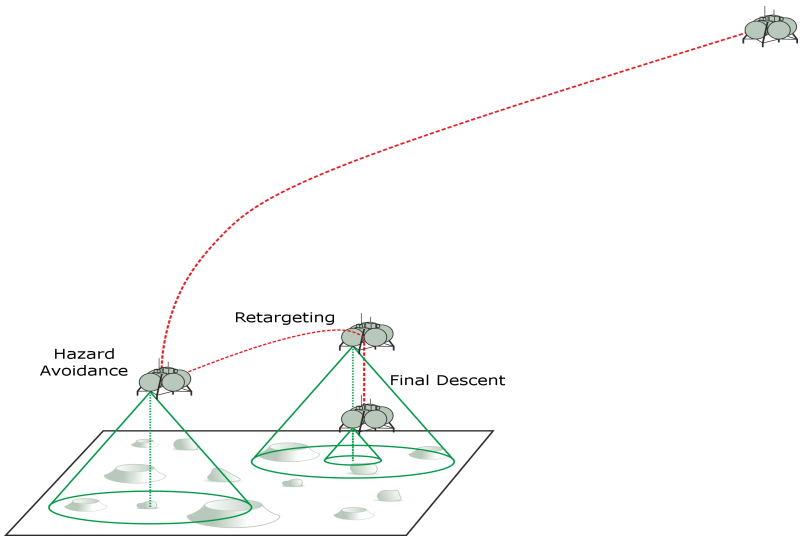


Figure 6.21: Hazard avoidance strategy

$$\frac{dr}{dt} = V \sin \gamma \tag{6.2}$$

$$\frac{d\theta}{dt} = \frac{V}{r} \cos \gamma \tag{6.3}$$

$$\frac{dV}{dt} = \frac{T}{m} \cos \alpha - g \sin \gamma \tag{6.4}$$

$$\frac{d\gamma}{dt} = \frac{T}{mV} \sin \alpha - \frac{g}{V} \cos \gamma \tag{6.5}$$

where r is the radial distance of the spacecraft from the centre of the Moon, V is the magnitude of the velocity of the spacecraft, γ is the flight-path angle, θ is the angle between the current radius vector and the radius vector at the start of powered descent, T is the magnitude of the thrust vector, α is the angle of attack, g is the magnitude of the gravitational acceleration, which is a function of r, and m is the current mass of the spacecraft. Figure 6.22 illustrates how the angles are defined, including the pitch angle β. Note that these equations describe the motion of the spacecraft in a vertical plane, i.e. there is no cross-range (out of plane) motion.

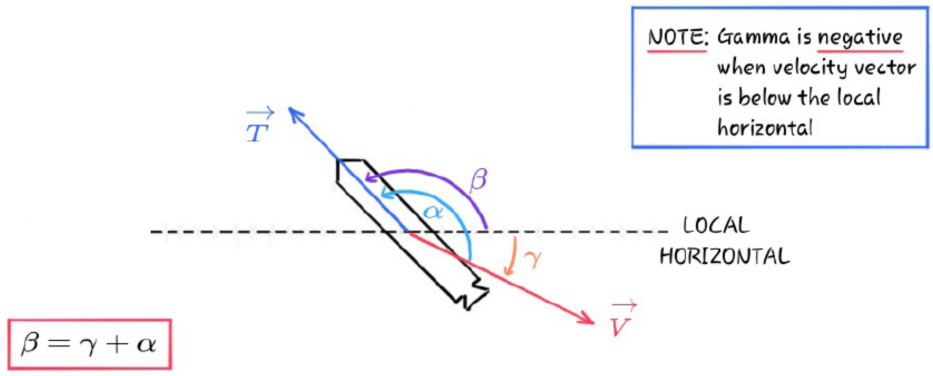


Figure 6.22: Diagram showing how the angles $\alpha, \beta,$ and γ are defined. Note: Angles are positive anti-clockwise.

Figures 6.23, 6.24 and 6.25 show the most pertinent parameters of the calculated trajectory, where Figure 6.23 shows the spacecraft altitude vs. ground distance, Figure 6.24 shows the altitude vs. time, and Figure 6.25 shows the pitch profile, which is the parameter that distinguishes each phase of the descent. The most important of these graphs is the altitude vs. ground distance plot, since the data used to produce this graph is the data that will be used to command the position of the camera for the capture of each image in the data set acquired from VISILAB.

6.4.3 Calculating the Scale Factor

Before the calculated trajectory can be applied to VISILAB it must be scaled down to a level appropriate for the dimensions of the VISILAB mock-up. Calculation of the scale factor can be achieved by using the calibration pattern to determine the perpendicular distance of the camera from the calibration pattern when the camera is in its initial position. This is the reason why a number of additional calibration images were acquired, with the camera in its starting position, during the calibration process described earlier.

Calculating Camera Position

The Matlab Camera Calibration Toolbox provides a function for calculating the extrinsic parameters of the camera using an image of a calibration pattern. Thus it provides an estimate of $\boldsymbol{R}$ and $\mathbf{t}$. Table 6.5 presents an example of the information returned by the extrinsics calculation. Even though a number of identical images have been captured with the camera in the starting position for a particular trajectory, the extrinsics calculation function can return quite variable results, which is likely to be due to image noise. Therefore, it is prudent to obtain an average of the extrinsic parameters over the set of all additional calibration images at the starting location. A weighted average can

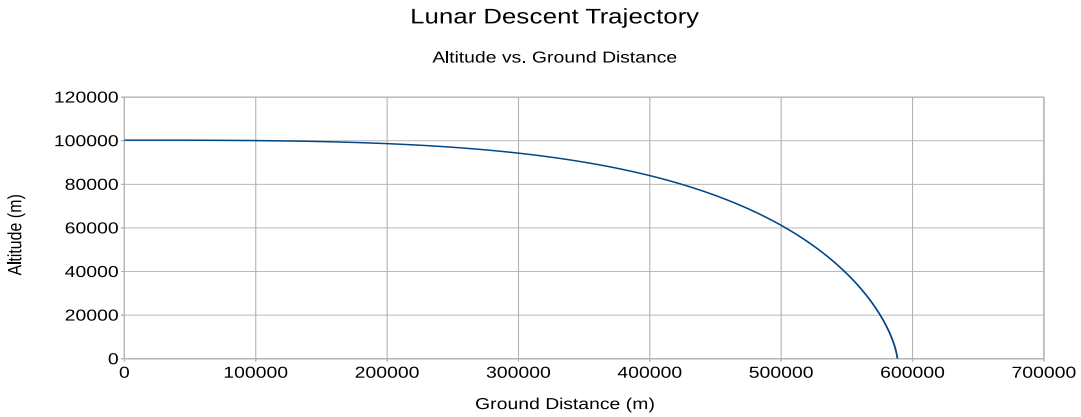


Figure 6.23: Altitude vs. ground distance plot

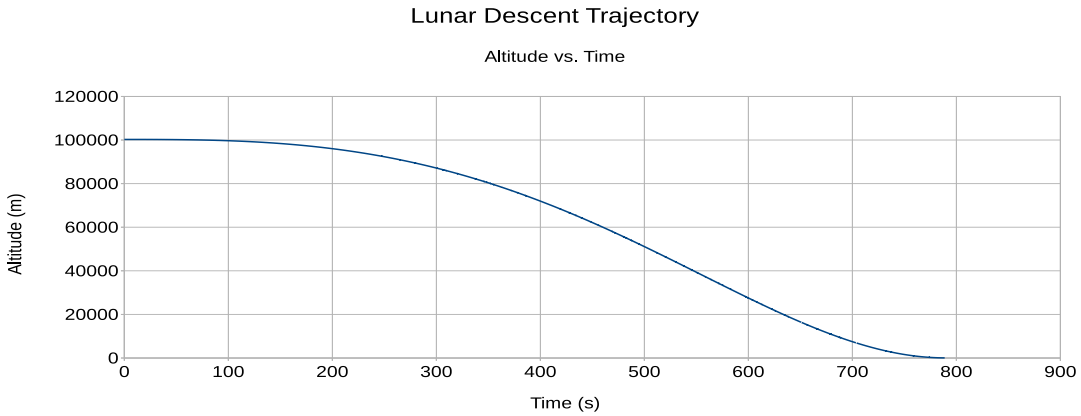


Figure 6.24: Altitude vs. time plot

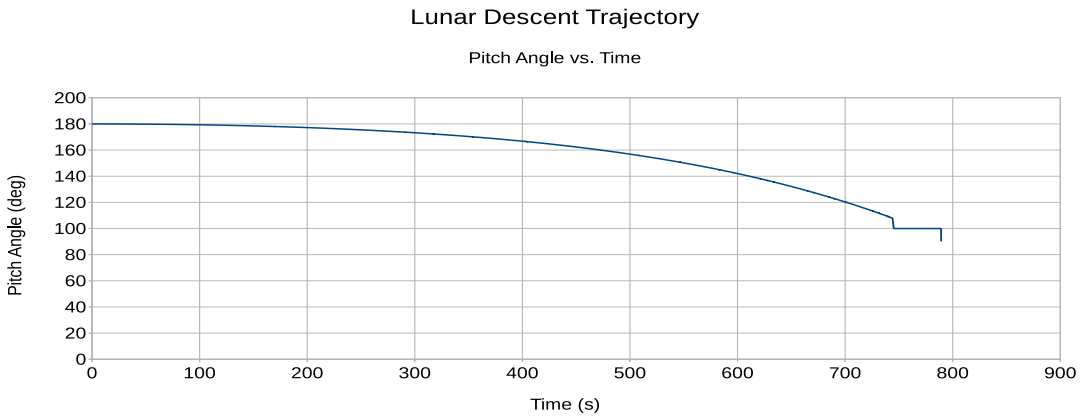


Figure 6.25: Pitch angle vs time plot

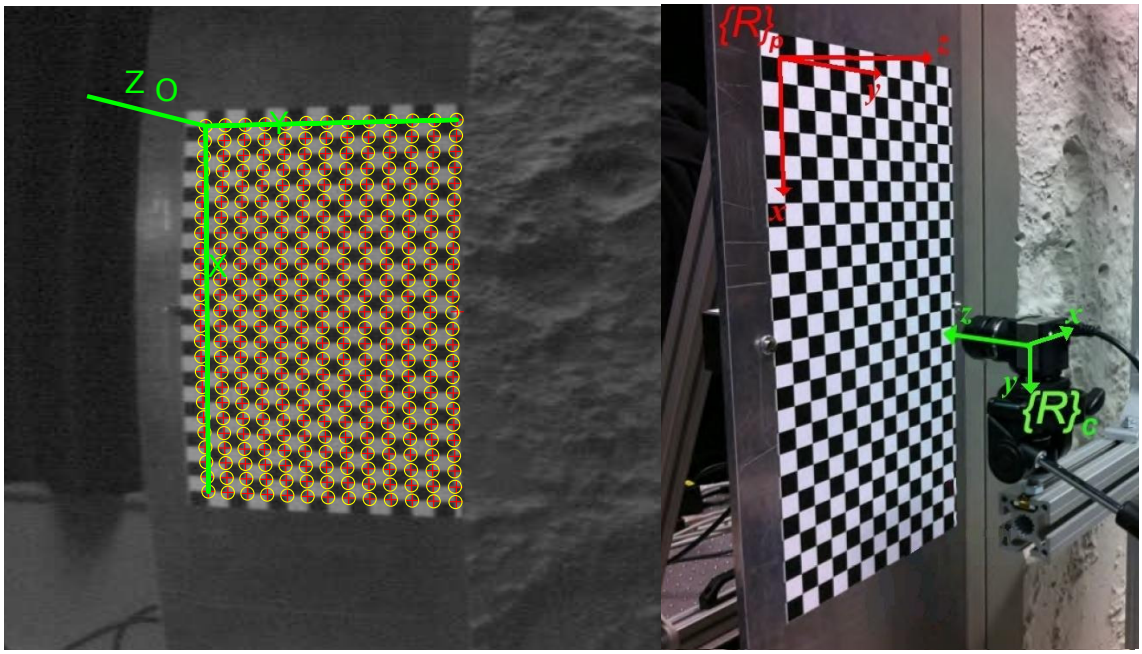


Figure 6.26: Reminder of different orientation of camera reference frame and calibration pattern frame. **Left:** Section of an additional calibration image (with camera in starting position) showing the calibration pattern and the corner extraction results from executing the extrinsics computation function. **Right:** Image showing the different orientations of the camera and calibration pattern coordinate frames — NOTE: the camera is not in the starting position in this image.

be computed by using the inverse magnitude of the error vector as the weight factor, thus giving more weight to the results that the extrinsic calculation function reports as being more accurate.

It is important to realise that the extrinsic parameters describe the position and orientation of the calibration pattern reference frame relative to the camera reference frame, in *camera frame coordinates*. However, since we will be using the origin of the calibration pattern reference frame as the origin of our world frame, we would actually like to describe the location and orientation of the camera reference frame with respect to the calibration pattern frame, in *calibration pattern coordinates*.

By closely examining the values in Table 6.5 and comparing the differences between

Extrinsic Parameters	**Estimated Values**
Translation Vector (mm)	$\begin{bmatrix} -345.718013 & -179.144593 & 878.356814 \end{bmatrix}$
Rotation Matrix	$\begin{bmatrix} -0.010425 & 0.999368 & -0.033972 \\ 0.999065 & 0.011836 & 0.041586 \\ 0.041962 & -0.033506 & -0.998557 \end{bmatrix}$
Error	$\begin{bmatrix} 0.20094 & 0.10965 \end{bmatrix}$

Table 6.5: Example of data returned by the extrinsics calculation function

the camera reference frame and the calibration pattern reference frame shown in Figure 6.26, and since when looking at the set-up from the front of the mock-up the camera is to the right and downwards of the calibration pattern coordinate frame, it is clear that the translation vector points from the origin of the camera reference frame to the origin of the calibration pattern frame. Similarly, it can be seen that the rotation matrix describes a rotation of the calibration pattern frame into the orientation of the camera frame, or alternatively pre-multiplying a vector expressed in calibration pattern coordinates by the rotation matrix would result in that same vector expressed in camera frame coordinates. Therefore, the translation vector describing the translation of the camera from the calibration pattern frame in calibration frame coordinates is given by:

$$\mathbf{t_p} = -\boldsymbol{R_{cp}}^T \mathbf{t_c} \tag{6.6}$$

where the minus sign is required to reverse the direction of the translation vector so that it points from the origin of the calibration pattern frame to the origin of the camera frame, and the rotation matrix is transposed so that it describes the transformation of a vector expressed in camera frame coordinates into the same vector expressed in calibration pattern coordinates. If we now go one step further and pre-multiply t_p by the rotation matrix

$$R_{wp} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.7}$$

i.e.

$$\mathbf{t_w} = \boldsymbol{R_{wp}}\, \mathbf{t_p} \tag{6.8}$$

we end up with the position vector, t_w, of the camera with respect to a world frame coordinate system oriented as shown in Figure 6.27. Therefore the position of the camera is now described in a coordinate system in which the x-y plane is co-planar with the mock-up surface, the z-x plane is co-planar with the x-y plane of the linear table, and the x axis also points in the same direction as that of the linear table. The motion of the camera in this new coordinate system is now trivially related to the motion in the linear table frame.

Scale Factor

Now that the position of the camera relative to the calibration pattern has been determined, the scale factor can be calculated by considering the perpendicular distance (z-coordinate) from the calibration pattern reference frame origin. This is achieved by dividing the perpendicular distance in mm, obtained from the extrinsic

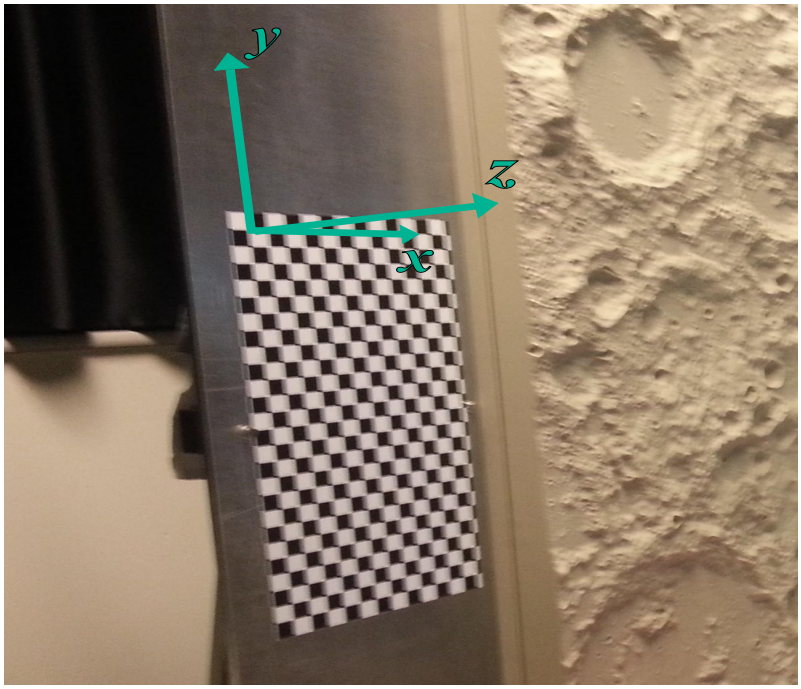


Figure 6.27: World frame coordinate system

parameters, by the initial altitude in m, computed from the trajectory calculation, to give the scale factor in mm/m:

$$s = \frac{z_{mm}}{h_m}. \tag{6.9}$$

The computed trajectory can now be scaled down to the dimensions of the VISILAB mock up by multiplying the altitude and ground distance, in m, for every image in the sequence, to give the altitude and ground distance in mm. The image dataset can now be collected by using these values in the linear table control software to set the position of the camera for each required image.

6.4.4 Collected Datasets

Using the approach described above, the following trajectories were calculated and image datasets were obtained, some of which are merely simplifications of the full lunar descent trajectory created by considering only vertical motion.

1. **Full Lunar Descent A:** This trajectory is based on the full lunar descent trajectory calculation described above. The calculated motion in the vertical and horizontal directions is fully simulated. However, the pitch angle is ignored and the camera points in the nadir direction at all times.

2. **Full Lunar Descent B:** Same as Full Lunar Descent A, except that in this case a constant look angle of 35 degrees (as defined above) is maintained throughout the descent.

3. **Vertical Descent Trajectory A:** Using the calculated altitude from the full lunar trajectory, the motion of the camera is purely in the vertical direction, i.e. we ignore the calculated horizontal motion. In this trajectory the camera maintains a nadir viewing direction.

4. **Vertical Descent Trajectory B:** Same as Vertical Descent Trajectory A, except that in this case a constant look angle of 35 degress is maintained during the descent.

5. **Vertical Descent Trajectory C:** The camera descends vertically with a constant speed of 100m/s, with a nadir viewing direction.

6.5 Tests on VISILAB Datasets

We now proceed to carry out some tests on these datasets collected from VISILAB. Before we do that, however, there are some issues that need to be addressed. Firstly, the VISILAB images, particularly those at the beginning of each sequence, contain a significant proportion of background objects within the field of view. This will result in the presence of extra feature points that do not fall upon the terrain surface mock-up (see Figure 6.28) and so their structure parameters should not be estimated, as these would not make sense in the context of this application. These features should be removed from the images, which could be achieved by applying a threshold to the x and y image coordinates, or, since we are not adding new features in any frames other than the first one, a simple case of choosing an appropriately large number of images of the sequence for which we want to track the feature points, such that at the end of the sequence these external features have left the field of view, will effectively eliminate these unwanted features. This second option, of course, requires that the features can be tracked for long enough to reach a point where all the unwanted features are excluded, but either way the end result is the same and both approaches are simple enough. The second option was used in the tests in this section (see Figure 6.28).

The next issue is somewhat more challenging. The tracked feature points will not have the same (x, y) image coordinates as their corresponding points in the ground truth DEM, therefore we must find a way of mapping the tracked feature points onto their corresponding points within the ground truth DEM in order to be able to make a comparison with the ground truth data and fully assess the performance of the SFM algorithm. In the interest of simplicity, and hopefully not at the expense of too much accuracy, we make use of a straightforward manual image registration tool that is available in Matlab in order to derive a transformation between the two image coordinate systems. This transformation can then be applied to the image coordinates of the

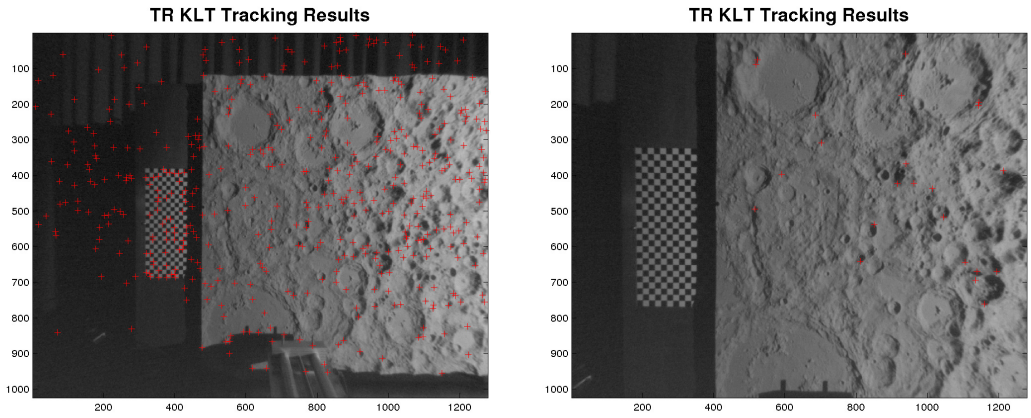


Figure 6.28: TR-KLT image features for the first image and last image for VISILAB Vertical Descent Trajectory A - In this case 23 features were remaining by image 160 of the sequence, which was the first point at which there were no background features present. These 23 features were then kept in all previous images and the rest were discarded in order to give a set of feature points lying entirely on the terrain surface. This method is used for all the VISILAB datasets used in the tests in this section.

feature points in the first image of the sequence in order to calculate the corresponding image coordinates of within the ground truth DEM. The procedure for determining the transformation function is now described.

The built-in Matlab tool that is employed for obtaining the mapping between the two image coordinate systems is called via the `cpselect()` Matlab function, using the first VISILAB image and the ground truth DEM image as input parameters. This presents the user with the interface shown in Figure 6.29, where the two images are shown side-by-side, with the full images shown at the bottom and zoomed views shown above. The user then examines each image to select a set of corresponding points. A set of around 65 matching points is shown in Figure 6.29 for the undistorted first image of Vertical Descent Trajectory C (each image data set will require this procedure to be repeated). These points can then be exported to the workspace and used to compute a projective transformation function that allows a different set of points (i.e. the tracked feature points) to be transformed from the VISILAB image to the DEM image. Figure 6.30 shows the results of this transformation applied to the original input points selected in the VISILAB image in order to provide a verification of the accuracy achieved. The top plot shows the full set of input points and the bottom plot shows a subset of these points after transformation into the ground truth DEM, alongside the manually determined corresponding points in the ground truth DEM. The view is zoomed to highlight the slight discrepancies in some of the points, but on the whole the result may be considered to be satisfactory. This will, however, be a source of errors in the sparse ground truth DEMs that will be generated to assess the performance of the SFM algorithm, and the errors may be slightly worse than those indicated in Figure 6.30 because here the transformation has been applied to those points that were used to calculate the transformation, whereas the tracked feature points will likely be an entirely

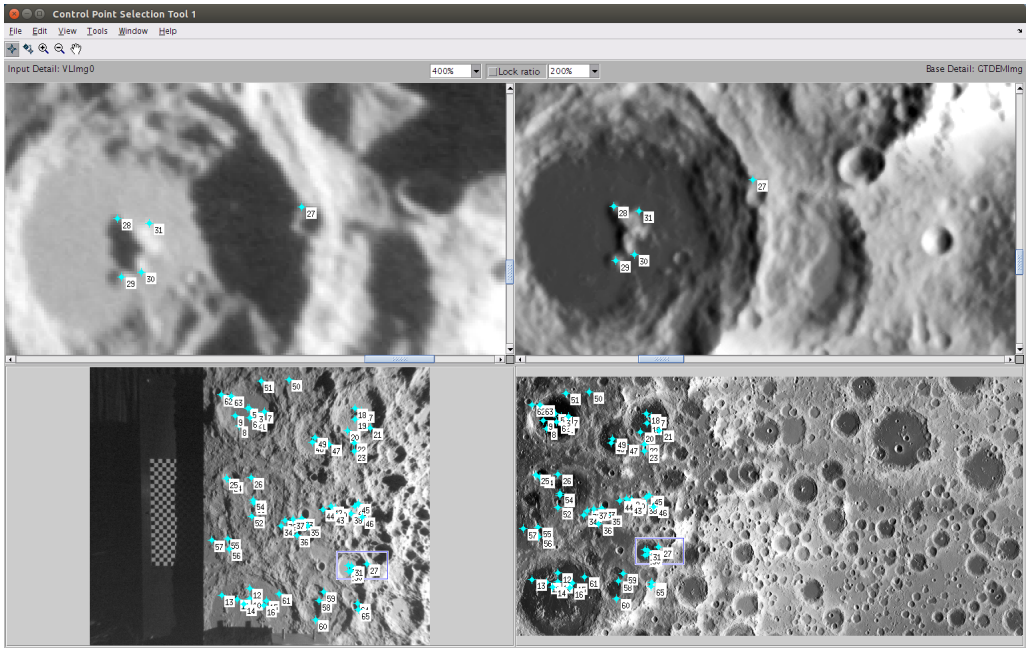


Figure 6.29: Matlab `cpselect()` tool user interface

different set of points, and so the transformation will be slightly less applicable, but hopefully still acceptable.

To assist in this manual determination of corresponding points, some processing of the supplied VISILAB ground truth DEM image file was required. The ground truth DEM, provided by the manufacturer of the VISILAB terrain, is provided in a 16-bit TIFF image, which is shown in Figure 6.31 (now in the correct orientation), which requires some preprocessing so that the image intensities actually represent physical elevation (instead of normalised elevation) with respect to the lowest point of the terrain surface (which would be a black pixel). The full height range of the surface is 44.730mm, thus a point that is 44.730mm above the lowest point would be represented by a white pixel. In order to achieve this scaling of the pixel grey levels the following transformation has to be applied:

$$I_{GT}(x,y) = \frac{44.73}{2^{16}} T_{GT}(x,y). \tag{6.10}$$

where $T_{GT}(x,y)$ is the original TIFF image intensity at point (x,y). The result of this transformation is presented in Figure 6.32, with an appropriate colour map to represent surface elevation. Unfortunately, if a grayscale colour map is applied to the transformed DEM, the resulting image, like the original TIFF image, does not have sufficient contrast to enable accurate and reliable manual point matching between the DEM and the VISILAB descent images. To combat this, we compute a Lambertian rendering of the transformed DEM image to produce an image that has a more physically realistic appearance with sufficient contrast to identify corresponding points. This rendered image is presented

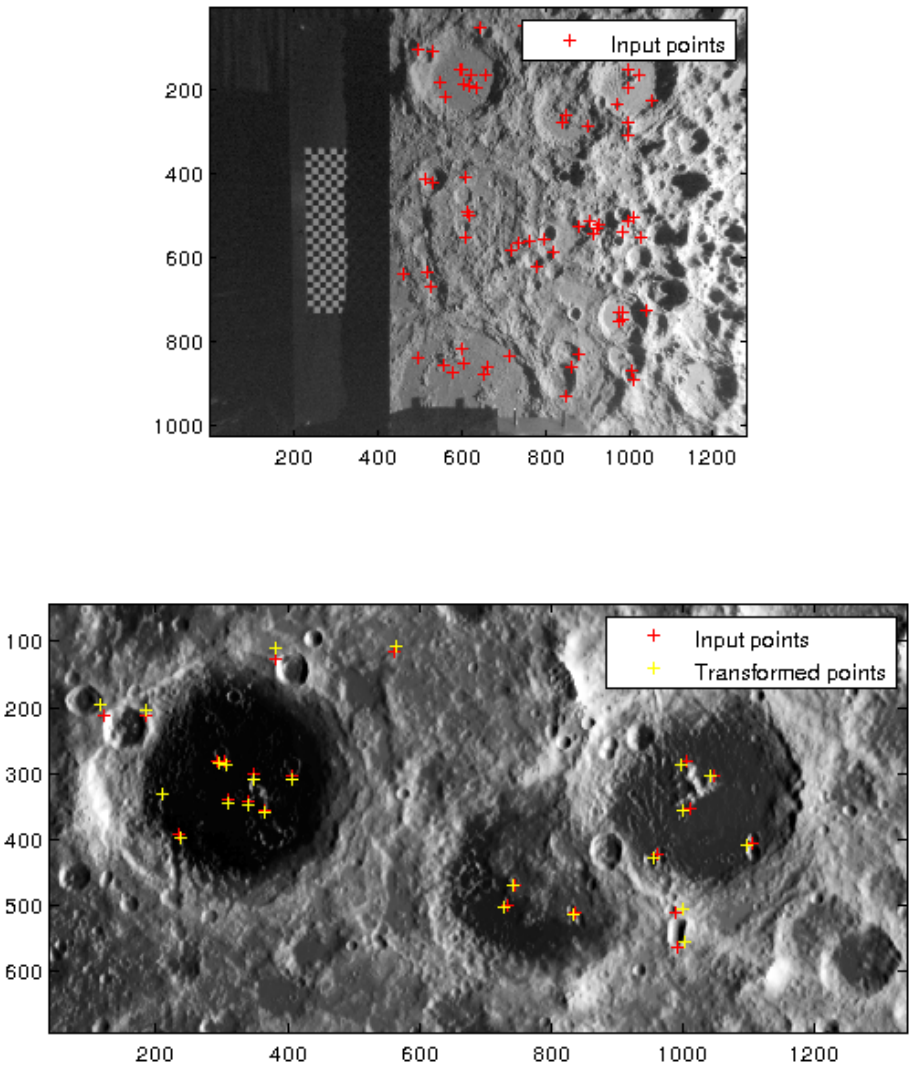


Figure 6.30: Verification of the transformation between VISILAB image (top) and ground truth DEM (bottom)

in Figure 6.33, and it is this image that was used to manually select matching points between the VISILAB descent image and the DEM in Figure 6.29.

An additional complication in using the VISILAB images derives from the subtle difference in how the camera motion is described and commanded with respect to the VISILAB apparatus and how the camera motion is described in the SFM algorithm. In VISILAB the motion of the camera was commanded with respect to the VISILAB world reference frame (see Figure 6.27), i.e. vertical camera motion is in the negative Z_w direction and horizontal motion is in the positive X_w direction. However, even when a nadir view was used in the VISILAB trajectories, the camera was positioned and oriented on the camera mount by hand so the orientation would only be approximately

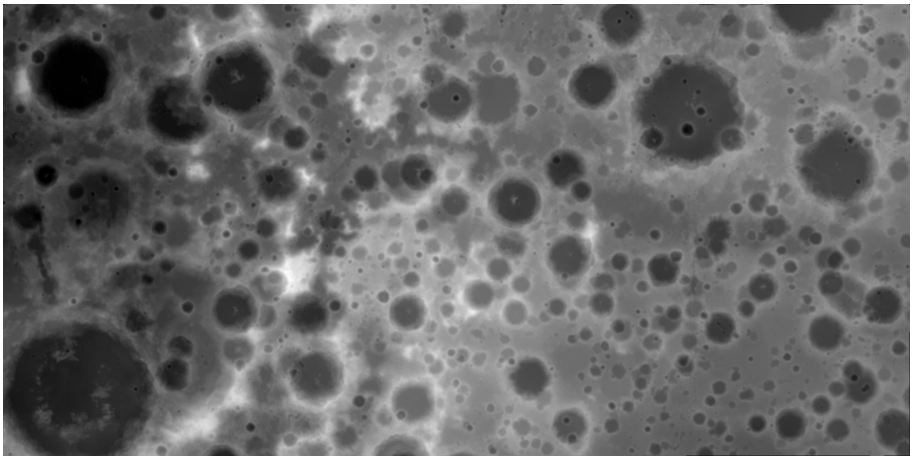


Figure 6.31: VISILAB ground truth DEM supplied in TIFF format (now in the correct orientation).

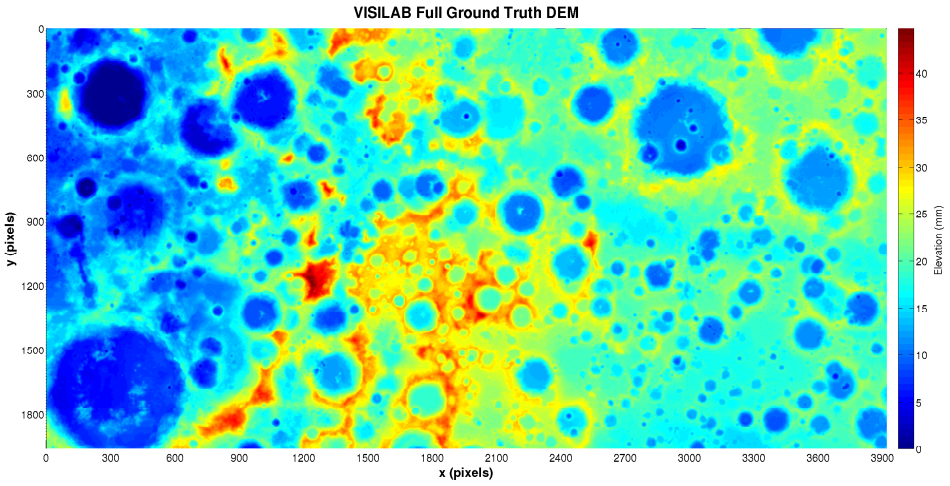


Figure 6.32: VISILAB full ground truth DEM

nadir, therefore the camera coordinate system and the VISILAB world frame are not necessarily in perfect alignment (parallel X-axes, and anti-parallel Z axes). This situation is illustrated in Figure 6.34, in which the misalignment is somewhat exaggerated. Figure 6.34 also indicates the component directions of motion of the camera during the capture of an image sequence, which are in the negative Z_w direction and (optionally) in the positive X_w direction of the VISILAB world reference coordinate system. The relationship between the calibration pattern reference frame and the camera frame is also shown in Figure 6.34, where it can be seen that the rotation matrix $\boldsymbol{R}_{cp}$, returned by the camera calibration toolbox during an extrinsics parameter estimation, describes a rotation of the calibration pattern coordinate frame,

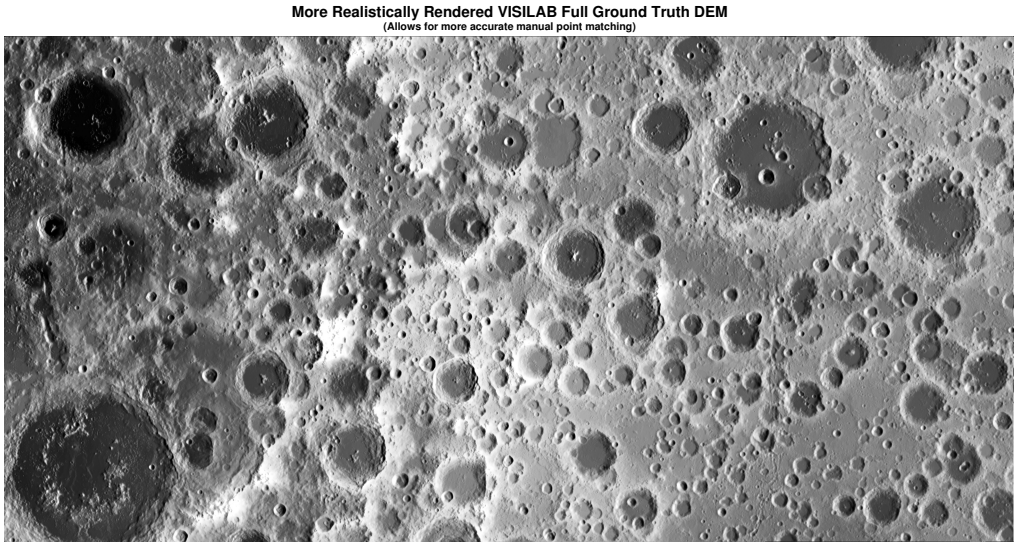


Figure 6.33: Lambertian rendering of VISILAB ground truth DEM, from which it is much easier to find corresponding points between the DEM and the VISILAB descent images

(O_p, X_p, Y_p, Z_p), into the orientation of the camera coordinate system, (O_c, X_c, Y_c, Z_c), and the translation vector, $\mathbf{T}_c$, describes the displacement of the coordinate system (O_p, X_p, Y_p, Z_p) from the origin of the camera coordinate system, in camera frame coordinates. Therefore, considering a point P_p in the calibration pattern coordinate system (denoted P_1 in Figure 6.34), where the coordinates of this point are denoted by $P_p = [X_p, Y_p, Z_p]^T$, this same point in the camera reference frame, $P_c = [X_c, Y_c, Z_c]^T$ can be computed via the following transformation:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \boldsymbol{R}_{cp} \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} + \mathbf{T}_c. \tag{6.11}$$

The relationship between the calibration pattern frame and the VISILAB world frame is given by the rotation matrix, $\boldsymbol{R}_{wp}$, which describes a rotation of the calibration pattern coordinate frame into the orientation of the VISILAB world frame, and is given by

$$\boldsymbol{R}_{wp} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{6.12}$$

Using Equations (6.11) and (6.12), the following transformation equation can be derived for converting points, such as P_2 in Figure 6.34, in the camera frame, to the VISILAB world reference frame:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \boldsymbol{R}_{wp}\boldsymbol{R}_{cp}^T \left(\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} - \begin{bmatrix} T_{X_c} \\ T_{Y_c} \\ T_{Z_c} \end{bmatrix} \right) \tag{6.13}$$

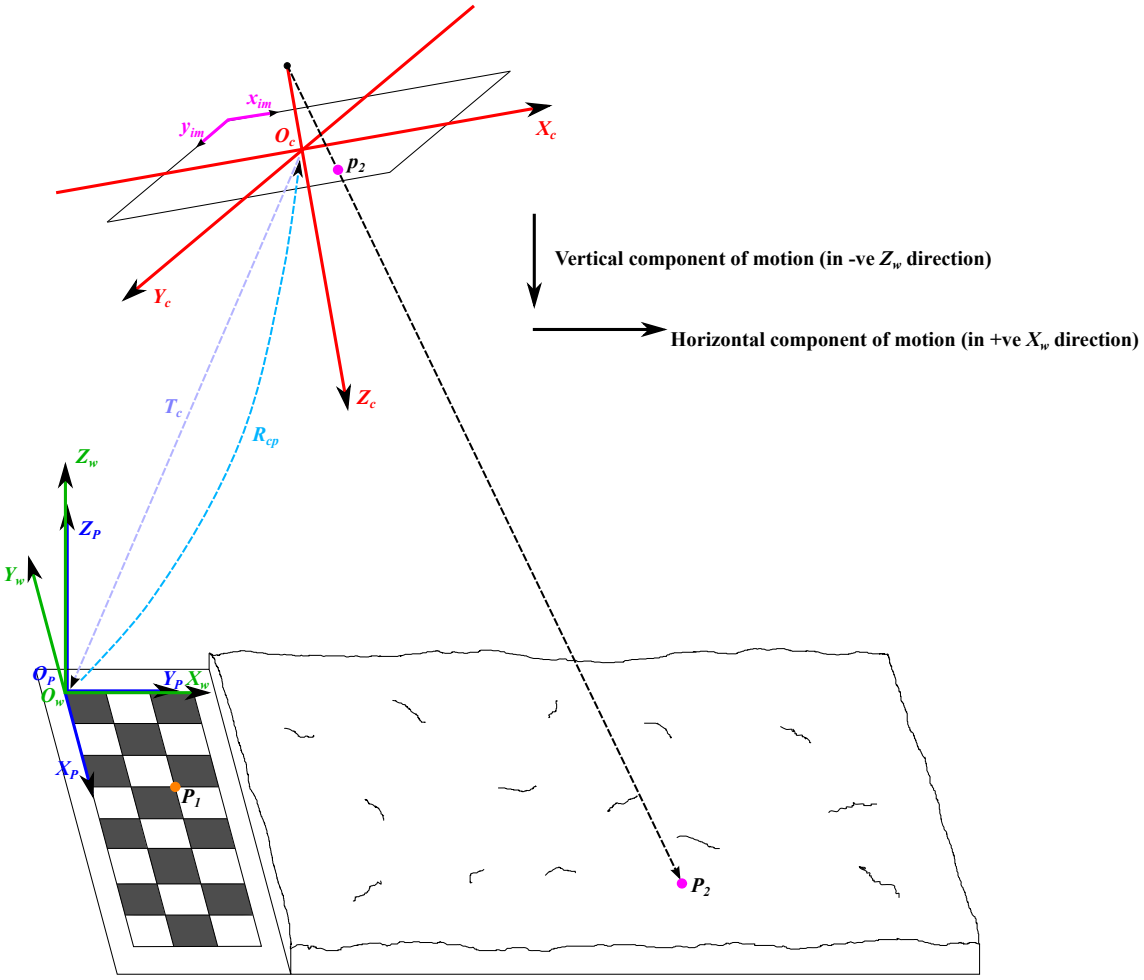


Figure 6.34: Relationship between VISILAB coordinate systems and the camera frame coordinate system. Also indicated are the component directions of motion of the camera, as commanded by the VISILAB software, which are in the negative Z_w direction and positive X_w direction.

Given that the calibration pattern coordinate system and the VISILAB world frame, have their origins in the same plane as the back plane of the VISILAB terrain surface, and that the lowest point on the terrain surface is 54.938mm above the back plane, applying Equation (6.13) and then subtracting $[0,\ 0,\ 54.938]^T$ from the resulting vector, enables the elevation of the feature points to be computed relative to a ground plane coincident with the lowest point of the terrain surface (and parallel to the backplane). This equation is used in constructing the DEMs estimated from the VISILAB image datasets, in which we are only interested in the Z components – the X and Y components produced by this equation unfortunately cannot be used because they do not correspond to anything meaningful in the ground truth DEM so they are discarded and instead we determine the x and y image coordinates of the tracked points within the ground truth DEM using the

manual mapping method described above.

To complete this analysis and describe the camera motion due to commanded motion with respect to the VISILAB world frame, we first rearrange Equation (6.13) so that it describes the transformation of a point from the VISILAB world frame to the camera frame:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \boldsymbol{R}_{cp}\boldsymbol{R}_{wp}^T \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + \begin{bmatrix} T_{X_c} \\ T_{Y_c} \\ T_{Z_c} \end{bmatrix}. \tag{6.14}$$

Now consider the point $[X_{c_0}, Y_{c_0}, Z_{c_0}]^T = [0, 0, 0]^T$ moving to a point $[X_{c_k}, Y_{c_k}, Z_{c_k}]^T = [\Delta X_c, \Delta Y_c, \Delta Z_c]^T$, e.g. the origin of the camera frame coordinate axes at the time, t_0, of capture of the first image moving to some other point at time $t = t_0 + k$ within the coordinate system of the original camera frame, due to the commanded motion $[X_{w_0}, Y_{w_0}, Z_{w_0}]^T \rightarrow [X_{w_k}, Y_{w_k}, Z_{w_k}]^T$ from $t_0 \rightarrow t_0 + k$. Inputting these values into Equation (6.14) for times t_0 and $t_0 + k$ and subtracting the two expressions gives

$$\begin{bmatrix} X_{c_k} \\ Y_{c_k} \\ Z_{c_k} \end{bmatrix} - \begin{bmatrix} X_{c_0} \\ Y_{c_0} \\ Z_{c_0} \end{bmatrix} = \boldsymbol{R}_{cp}\boldsymbol{R}_{wp}^T \left(\begin{bmatrix} X_{w_k} \\ Y_{w_k} \\ Z_{w_k} \end{bmatrix} - \begin{bmatrix} X_{w_0} \\ Y_{w_0} \\ Z_{w_0} \end{bmatrix} + \begin{bmatrix} T_{X_{c_k}} \\ T_{Y_{c_k}} \\ T_{Z_{c_k}} \end{bmatrix} - \begin{bmatrix} T_{X_{c_0}} \\ T_{Y_{c_0}} \\ T_{Z_{c_0}} \end{bmatrix} \right).$$

In the SFM algorithm, all motion (and structure) is described with respect to the coordinate system defined by the position and orientation of the camera reference frame at the time that the first image was captured, therefore if $[X_{c_0}, Y_{c_0}, Z_{c_0}]^T = [0, 0, 0]^T$ is the origin of the camera reference frame at the time of the first image, then the point $[X_{c_k}, Y_{c_k}, Z_{c_k}]$ is the origin at $t_0 + k$, but expressed in the camera reference frame at t_0. The translation vectors are also both in this same reference coordinate system and are in fact the same, so they cancel out. Therefore, the position of the origin of the camera frame at time $t_0 + k$ in the coordinate system of the original image camera frame is given by the following expression:

$$\begin{bmatrix} X_{c_k} \\ Y_{c_k} \\ Z_{c_k} \end{bmatrix} = \boldsymbol{R}_{cp}\boldsymbol{R}_{wp}^T \left(\begin{bmatrix} X_{w_k} \\ Y_{w_k} \\ Z_{w_k} \end{bmatrix} - \begin{bmatrix} X_{w_0} \\ Y_{w_0} \\ Z_{w_0} \end{bmatrix} \right) = \boldsymbol{R}_{cp}\boldsymbol{R}_{wp}^T \begin{bmatrix} +|\Delta X_{w_k}| \\ 0 \\ -|\Delta Z_{w_k}| \end{bmatrix}, \tag{6.15}$$

where we have emphasised the signs of the ΔX_w and ΔZ_w components to indicate the directions of motion as commanded by the VISILAB software. Note also that ΔY_w is always zero because this direction of motion is not easily controlled using the current configuration of VISILAB, therefore it was kept constant at all times.

A final point that needs to be highlighted is that all images for each dataset need to be undistorted prior to their use in the feature tracking algorithm, which in the tests that follow is the TR-KLT algorithm that was described in Chapter 3.

6.5.1 Test on VISILAB Vertical Descent Trajectory C

We begin by testing the performance of the PSO initialised MS-SRUKF algorithm, with TR-KLT feature tracking, on the VISILAB Vertical Descent Trajectory C, in which the

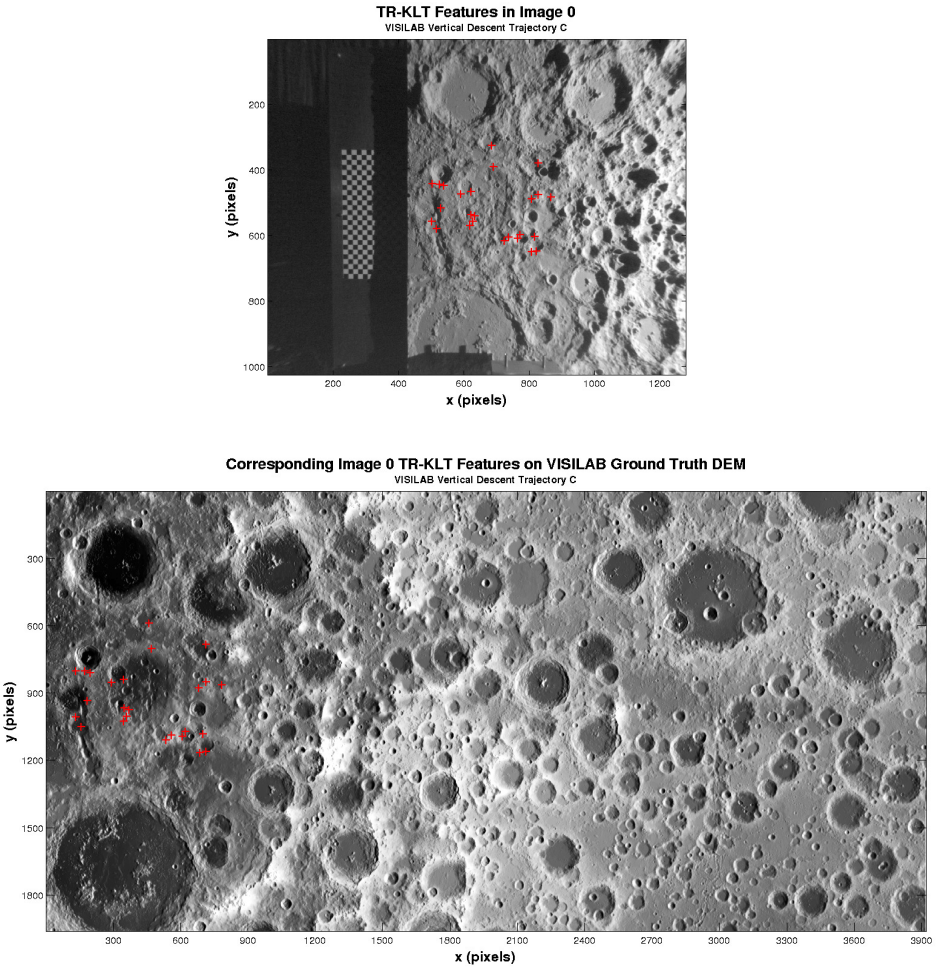


Figure 6.35: Top: TR-KLT feature point distribution in first image of VISILAB Vertical Descent Trajectory C - 25 feature points were tracked over 280 images; Bottom: Corresponding first image feature point locations in the VISILAB ground truth DEM

spacecraft begins at a 2000m altitude and undergoes a pure vertical descent at a constant velocity of 100m/s, with a nadir view. This trajectory is the most similar to the previous tests using PANGU images, and so a comparison between these results and the previous results can be more readily made.

For this test, and all other tests on the VISILAB data sets, we only select features in the first image of the sequence and then attempt to track these features over all subsequent images using the TR-KLT feature tracker. To ensure that only features that lie on the VISILAB terrain surface are used in the SFM algorithm, we track the features up to a point where all background features have either left the field of view or become lost by some other means, as discussed above. For this particular test, 25 terrain features were tracked over a 280 image sequence. The distribution of these feature points in the first image of the sequence is presented in Figure 6.35, which therefore

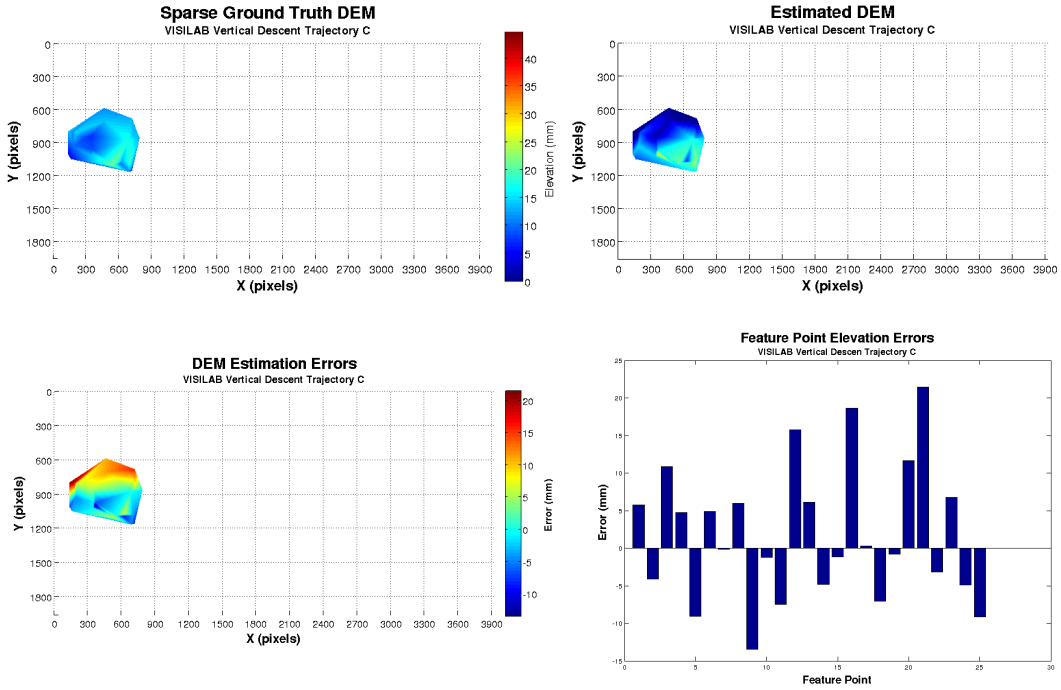


Figure 6.36: SFM Results for VISILAB Vertical Descent Trajectory C – Top: Sparse ground truth DEM (left) corresponding to the estimated DEM (right) with interpolation between the tracked feature points; Bottom: DEM elevation errors (left) and feature point elevation errors (right)

also indicates the density of the estimated DEM. The fact that these 25 features could be tracked for 280 images is evidence that the TR-KLT feature tracker performs much better on real images than on the synthetic PANGU image sequences in which similar numbers of features could only be tracked for around 100–120 images.

Figure 6.36 presents the results obtained from the MS-SRUKF algorithm using the 280 images from the VISILAB Vertical Descent Trajectory C image dataset, over which 25 features were tracked from the first image, without any feature renewal. The top part of Figure 6.36 presents the estimated DEM (right) alongside a sparse ground truth DEM (left) constructed by sampling the full resolution VISILAB ground truth DEM at the locations of the tracked feature points in the first image of the sequence. From this it can be seen that, for the most part, at this level of sparsity, the surface has been estimated to a reasonably high level of accuracy. This is especially the case in the lower $\sim {}^{2}/_{3}$ of the DEM, where, from the DEM surface error plot on the bottom left, it can be seen that the errors are very close to zero in this region. The top ${}^{1}/_{3}$ of the estimated DEM, on the other hand, contains significant error. From the error DEM it can be seen that the estimated DEM is approximately 10–20mm lower in elevation than the sparse ground truth. The bar chart on the bottom right of Figure 6.36 shows more clearly the errors for each individual feature point, from which it can be seen that the majority of the feature points (52%) are below 6mm, and the peak error is approximately 20mm. With further

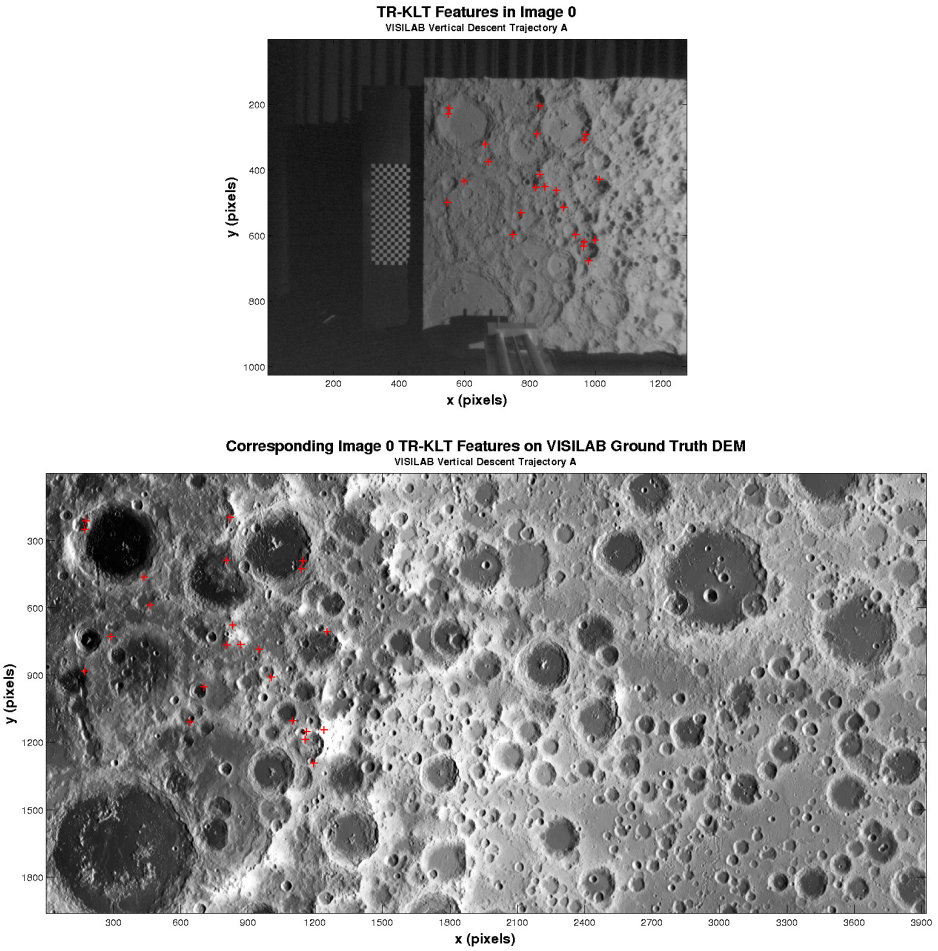


Figure 6.37: Top: TR-KLT feature point distribution in first image of VISILAB Vertical Descent Trajectory A - 23 feature points were tracked over 160 images; Bottom: Corresponding first image feature point locations in the VISILAB ground truth DEM

analysis, the MS-SRUKF algorithm for this dataset, resulted in 12% of the feature points being estimated to within ± 1mm, 20% of the features were estimated to within ± 3mm, and 44% of the features were estimated to within ± 4mm, and resulted in an overall RMS error of approximately 9mm. Unfortunately, since the entire VISILAB terrain has a total elevation range of around 45mm, the maximum error of 20mm is very significant, so it is clear that further work is required to achieve a greater accuracy.

6.5.2 Test on VISILAB Vertical Descent Trajectory A

In this case we test the performance of the MS-SRUKF algorithm on VISILAB Vertical Descent Trajectory A, in which the camera descends vertically towards the VISILAB terrain surface with a nadir view and where the camera motion is produced from use of

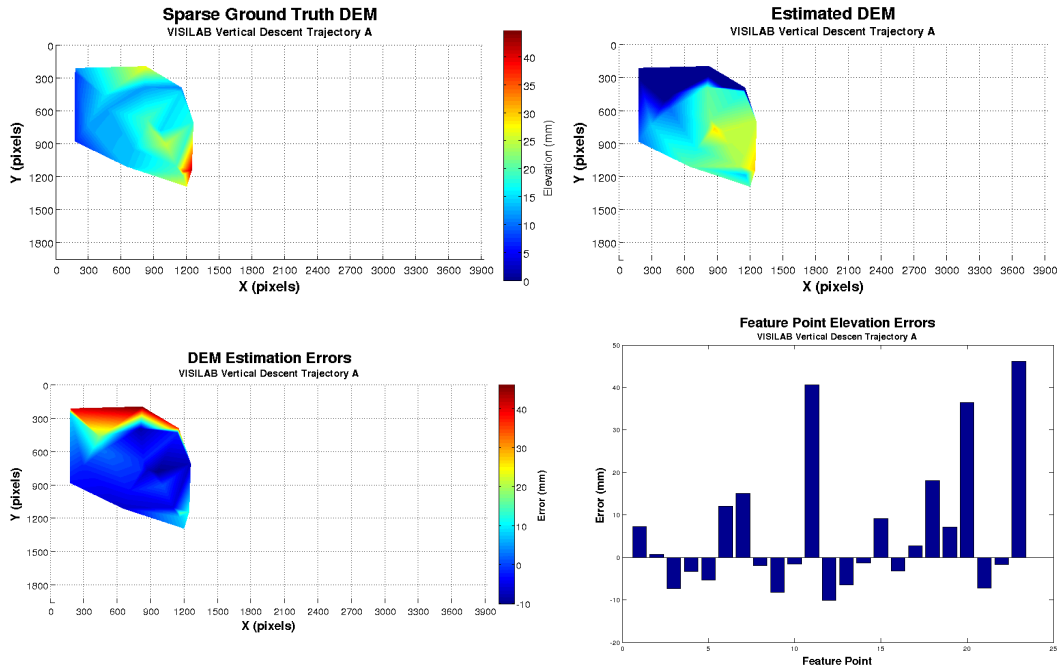


Figure 6.38: SFM Results for VISILAB Vertical Descent Trajectory A – Top: Sparse ground truth DEM (left) corresponding to the estimated DEM (right) with interpolation between the tracked feature points; Bottom: DEM elevation errors (left) and feature point elevation errors (right)

only the vertical component of motion from the Full Lunar Descent Trajectory. In this way, this test form the next logical step towards a full Lunar descent trajectory from the simple constant velocity vertical descent trajectories of the previous test and the PANGU tests.

Figure 6.37 shows the distribution of the TR-KLT feature points extracted from the first image that were able to be tracked in every image up until all the features produced from background objects had left the field of view or had been lost by other means. Here there were 23 features tracked for 160 images, and in this case they are more spread out over the surface, which means that the density of the DEM is slightly lower than before. The top plot of Figure 6.37 shows the distribution of the 23 feature points in the first image of the sequence for VISILAB Vertical Descent Trajectory A, and the bottom plot show their corresponding locations in the VISILAB ground truth DEM image as determined by applying the calculated projective transformation from the manual point mapping procedure.

The results obtained from using these feature points in the MS-SRUKF SFM algorithm are presented, along with error analysis, in Figure 6.38. The sparse ground truth DEM (top-left) is presented alongside the estimated DEM (top-right), from which it can be seen that reasonable performance has been obtained from the SFM algorithm for the lower approximately $4/5$ of the estimated region of the VISILAB surface, although the small high elevation region at the bottom right corner is somewhat under estimated by around 10mm. This is reflected in the plot of DEM elevation errors in the bottom-left

of Figure 6.38, which indeed shows around a 10mm error for this small region. For the rest of the lower $4/5$ region of the DEM the errors appear to be very close to zero on the whole, however there are a few points that appear to have an elevation slightly below that of the ground truth. This is further confirmed by analysing the bar chart plot of the errors for each individual tracked point, in which it can be seen that there are a number of points that are close to zero error. However, there are 3 points that have very high errors of around 40–45mm, which is particularly significant given that the full range of elevation over the entire VISILAB surface is 44.73mm. Overall the results from this test are not as good as the results obtained for Vertical Descent Trajectory C, which can be seen from the fact that the RMS error here is 16.65mm whereas for Trajectory C it was 9.03mm. The reason for this is likely due to it being heavily skewed due to those points that are estimated extremely poorly, because if we reject these points the RMS error falls to 7.97mm, which is in fact an improvement over the results obtained in Trajectory C. This therefore suggests that the SFM algorithm could be improved by incorporating some sort of outlier rejection strategy, such as RANSAC applied to the feature detector, which would lead to the rejection of any points that exhibit image motion characteristics that are too dissimilar from the other features. Whether this would be sufficient to remove all those features that would result in a poor structure estimate would need to be rigorously examined in any future work. Alternatively, it may be possible to directly apply a related method upon the actual structure estimates themselves in order to detect and reject any features that have radically different structure estimates from the other feature points. Deriving such an outlier rejection scheme is left to future work as this would likely require an extensive search of the available literature to identify a suitable approach. Without rejecting these extreme structure estimates, the majority of the feature points (61%) are now estimated to within ± 7mm, which is again a reflection of the reduced performance in this test since previously over 50% of the features were within ± 5mm, however the influence of the three extremely erroneous feature points is significant in this result.

6.5.3 Test on VISILAB Full Lunar Descent Trajectory A

We now carry out a test of the MS-SRUKF SFM algorithm on the full VISILAB Lunar Descent Trajectory A, which consists of both vertical and horizontal camera motion based on a powered Lunar descent trajectory scaled down to the dimension of the VISILAB equipment, as described earlier. For this trajectory, the camera maintains a constant nadir viewing angle.

Figure 6.39 shows the distribution of the tracked feature points in the first image of the sequence (top) and the corresponding locations of these feature points within the VISILAB ground truth DEM image (bottom). In this case 30 features were remaining after 220 images, which was the point at which all background features had disappeared.

Figure 6.40 presents the SFM results and error analysis obtained from this dataset using this set of feature points. Along the top of the figure are the sparse ground truth DEM (left) and the corresponding estimated DEM (right). Here we can see that large portions of the DEM appear to have been estimated to a close agreement with the ground truth, suggesting that any of the feature points have been estimated to high accuracy.

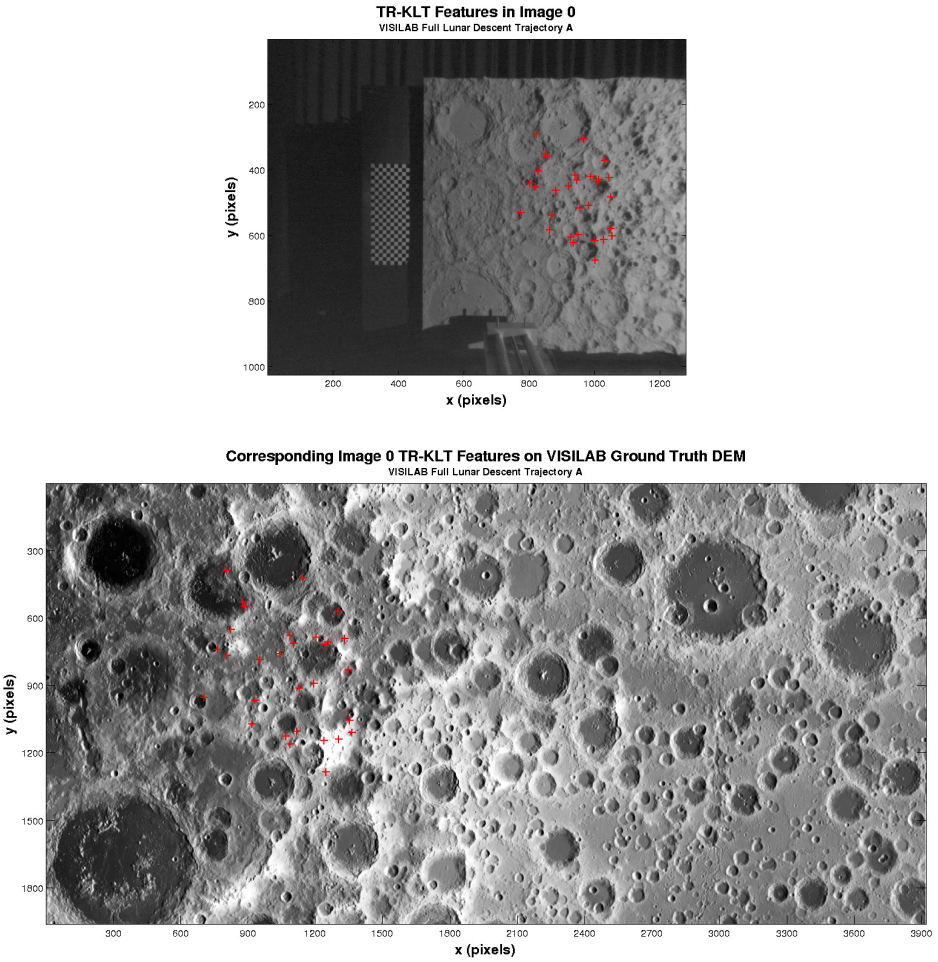


Figure 6.39: Top: TR-KLT feature point distribution in first image of VISILAB Full Lunar Descent Trajectory A - 30 feature points were tracked over 220 images; Bottom: Corresponding first image feature point locations in the VISILAB ground truth DEM

However, some discrepancies are clearly visible in both the upper-middle region, where a very low elevation has been estimated, and to the middle-right/lower right region, where a relatively high elevation has been estimated. In the sparse ground truth DEM there is a region stretching from the mid-right to the bottom-right that has a reasonably high elevation, and is therefore coloured red, that appears to form an arrow-like shape. The left half of this arrow-like shape is clearly visible in the estimated DEM, thus in terms of shape this is a positive result even though the elevation underestimated by around 10mm, however the right half of this arrow-like shape is not very apparent in the estimated DEM as it is obscured by an erroneous neighbouring region that has been estimated to have a similar elevation to the arrow-like region. There is also a very small region in the bottom left that erroneously has extremely low elevation, when in fact it should have a high elevation. These observations are reflected in the DEM error surface plot, which

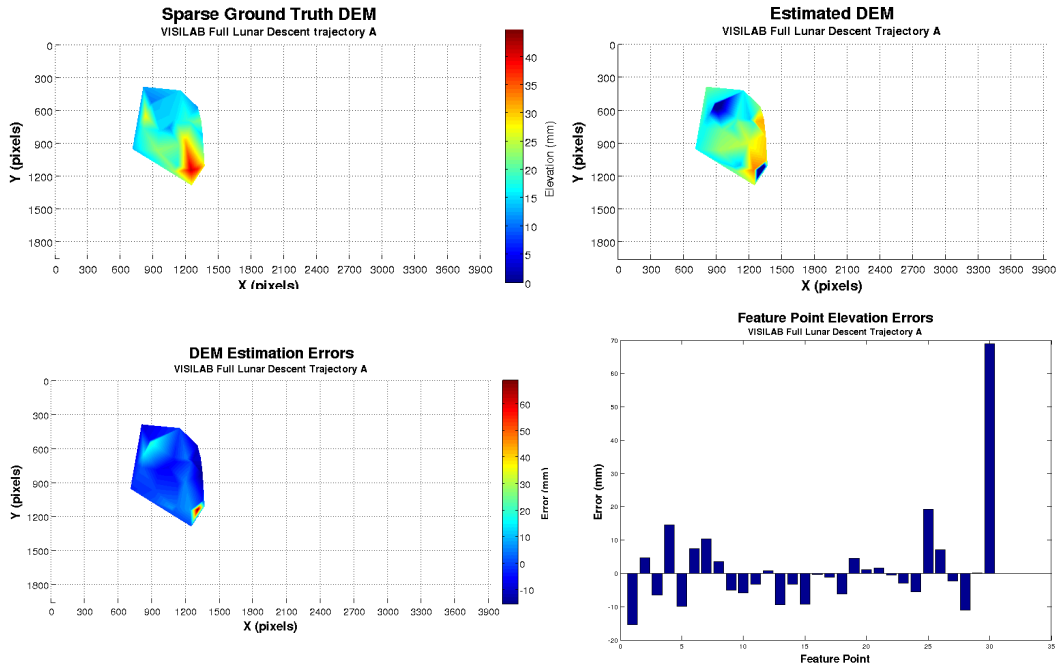


Figure 6.40: SFM Results for VISILAB Full Lunar Descent Trajectory A – Top: Sparse ground truth DEM (left) corresponding to the estimated DEM (right) with interpolation between the tracked feature points; Bottom: DEM elevation errors (left) and feature point elevation errors (right)

appears to show that most of the surface has small errors, apart from in the regions noted above. Since this error plot is computed by subtracting the estimated (interpolated) DEM from the sparse (interpolated) ground truth DEM, when the true surface has a high elevation but the estimated surface has a low elevation, the error is positive, and when the true surface has a low elevation but the estimated surface has a high elevation, then the error is negative. Therefore, the small region of extreme error in the bottom-right shows a high positive error since the estimated surface has a very low elevation here when in fact it should be a high elevation. We can also see a region of negative error in a strip along the right edge of the surface (right of the arrow-shaped region), which is where a mid-to-high elevation was estimated when it actually should have been a lower elevation. On the upper-left there is a region of approximately $+15$mm error, resulting from the small deep-blue region that resulted in an elevation about 20mm lower than it should have been. These observations are supported by the bar plot (bottom-right) of errors for each individual feature point. Of particular concern is the extremely large positive error of nearly 70mm, corresponding to the small highly erroneous region in the bottom-right of the estimated DEM, which is very significant given that the entire surface only has a full elevation range of 44.73mm. This strongly skews the RMS error to 14.64mm, whereas if this feature point could have been rejected by an outlier removal scheme then the RMS error would be reduced to 7.62mm. Additional analysis of the errors shows that exactly 50% of the surface is estimated to within ± 5mm, and so a similar level of accuracy is

achieved in this test to that of Vertical Descent Trajectory C.

6.5.4 Conclusions from VISILAB Tests

In this section, three tests have been carried out using the collected VISILAB datasets. On the whole the results have been positive, with many of the features estimated to high accuracy. However, a number of discrepancies have been observed in which a handful of the tracked feature points resulted in highly inaccurate structure estimates. These outliers resulted in an overall reduction in accuracy than was observed in previous tests using PANGU images. It is clear that improvements could be made with the incorporation of a strategy for detecting and removing outliers so that these extreme feature points could be culled from the set of tracked features, and therefore improving the overall accuracy of the estimated terrain surface. Precisely what form this outlier rejection scheme would take is not known at this stage and so further research is needed. One possibility, however, could be the use of a RANSAC algorithm acting only on the observed image motion of the features during the feature tracking stage, wherein any features that exhibit a dissimilar motion behaviour to the rest of the features will be identified and removed. Whether this would be an effective means of removing those features that would go on to produce the highly erroneous structure estimates in the SFM algorithm would need to be the subject of an extensive investigation. The problem may also be down to a suboptimal set of tuning parameters, since for the tests carried out in this chapter the PSO filter initialisation did not result in satisfactory SFM results, despite achieving a very low cost, and consequently a significant amount of manual refinement of the PSO derived tuning parameters was required. This may indicate a problem with the derivation of the cost function and thus indicate that a slightly different formulation may be required since a simple matching of covariances did not appear to be sufficient. However, the PSO initialisation appeared to work much better for the synthetic PANGU image sequences, and so in those cases the cost function seemed to be capable of producing satisfactory tuning parameters. This may actually also indicate that the difficulty in PSO tuning on these real image data sets could be due to the presence of outliers that were not present in the PANGU tests. An increase in difficulty, to some extent, would be expected when using real imagery as opposed to synthetic imagery where things are much more straightforward and controlled, for example, with synthetic images, the complexity of dealing with image distortions is not present, which can introduce a significant source of errors when undistorting images if the camera calibration is not entirely accurate. It is therefore clear that additional mechanisms may need to be incorporated to increase the robustness of the entire feature tracking and SFM algorithm.

We should also point out that only those image datasets in which the camera had a nadir viewing angle were used in the above tests. Unfortunately, when the camera had a 35 degree look angle, the calibration pattern was not fully visible when the camera was in its starting position. In this case, it was deemed reasonable to assume that the same scaling could be applied to the trajectory motion parameters and so it was not considered vital to be able to see the calibration pattern in order to obtain the extrinsic parameters. However, while it was reasonable to assume the scaling would not change, due to time restrictions during the collecting of the datasets, the full importance of being able to obtain

the extrinsics parameters, for determining the orientation of the camera frame so that the motion could be converted to the camera frame in order to predict the feature point measurements in the filter, was overlooked. This could be overcome by removing the calibration pattern from its default position and temporarily placing it in front of the terrain so that it is within the field of view of the camera. The extrinsics parameters could then be estimated with respect to this location, and then due to the precision placement of objects that is provided by the optical table, the resulting extrinsics parameters could be transformed back into the original calibration pattern frame and the analysis could then proceed as normal. Obtaining results from the two remaining datasets is therefore left as further work.

7 SHAPE FROM SHADING

In this chapter we investigate some techniques for estimating the shape of a surface using the shading information that is present in the images captured from the on-board descent camera. Specifically we look at an approach known simply as shape from shading (SFS), which uses a single image to estimate the shape of the terrain; a technique known as photometric stereo, which works on the same principle but uses three images taken with different light source directions in order to bypass some of the assumptions and approximations that must be made to estimate the surface shape when using only a single image. Additionally an extremely simple method of directly estimating slope from shading information is investigated, which bypasses many of the assumptions inherent in SFS, and sets of hazard maps are produced to indicate areas that would be too dangerous on which to land. While somewhat promising, this simple method unfortunately misses a number of unsafe regions in comparison to ground truth data, and so finally a much more capable method is investigated and analysed. This final method is a very modern and sophisticated shape from shading technique that uses prior knowledge of natural image statistics in order to recover dense shape as well as variable surface albedo from shading information and also allows low resolution initial shape estimates to be incorporated. Consequently, this sophisticated method could be used in combining the previously obtained sparse SFM structure estimates with a shape from shading method, in order to provide fully scaled and dense DEMs of the landing site region. These SFS techniques are focussed solely on the production of DEMs of the landing site and the ultimate aim is to develop a technique that would supplement, incorporate and improve the DEMS generated from the structure from motion approaches in the previous chapters.

7.1 Background and Motivation

The goal of shape from shading algorithms is to derive a 3D scene description from one or more 2D images. There are numerous ways in which this description can be expressed, such as depth $Z(x, y)$, surface normal (n_x, n_y, n_z), surface gradient (p, q), and surface slant, ϕ, and tilt, θ [150]. Shape from shading deals with the recovery of such a description of the surface using the gradual variation of shading in an image, by considering the way in which light falls upon and is reflected by the surface. The most common way of describing the image formation process is through the use of the Lambertian reflectance model, in which the observed gray level at a pixel in the image depends on the light source direction and the surface normal [150]. It must be noted, however, that the use of the Lambertian model is often too simplistic because real images do not always contain features that can be completely described by Lambertian reflectance; there are often other features present such as shadows, and specular reflectance, for example. Even if Lambertian reflectance can be used to adequately describe the entire surface visible

in an image, the SFS problem is still difficult to solve. The reason for this is that, for example, if we wish to describe the surface in terms of the surface normal, we have one linear equation with three unknowns, or if we are using surface gradient then there is one non-linear equation with two unknowns. Therefore, in order to proceed, a number of additional constraints must be applied that enable a unique solution [150].

The above mentioned issues that are encountered in shape from shading problems have lead to a number of different methods and algorithms being proposed in the literature, each of which has differing strengths and weaknesses in differing situations. There are four main categories of shape from shading approaches: minimisation methods, propagation methods, local methods, and linear methods. Minimisation methods work by constructing an energy function that contains some constraints to overcome the ambiguities that arise from having an under constrained system (e.g. a brightness constraint that ensures the reconstructed shape produces the same brightness as the input image, and a smoothness constraint to ensure a smooth surface reconstruction see [151]), and then minimising this energy function to obtain the shape solution [150]. Propagation approaches work by constructing a set of initial surface curves around the neighbourhood of some special points in the image (e.g. singular points) using an initial assumption such as a local sphericity and then propagating the solution outwards using the direction of intensity gradients. The first shape from shading algorithm, developed by Horn [152], was a propagation approach that used the method of characteristics to propagate the solution along lines known as characteristic strips. A characteristic strip is a line in the image along which the surface depth and orientation can be computed if these quantities are known at the starting point of the line [150]. The starting points of the line are the initial surface curves around the singular points as mentioned above, therefore the starting points come from an assumption about the surface which may not necessarily hold. An overview of this first method is presented in Figure 7.1 to provide an example of how shape from shading is computed. This particular algorithm was chosen as an example because it was the foundation that inspired all the subsequent SFS techniques, even though the more recent algorithms can be quite different to this original approach. Local approaches derive shape based on assumptions about surface shape [150]. An example of a local approach is the work of Pentland [153], which recovered shape information from the intensity, and its first and second derivatives, by assuming that the surface is locally spherical at each point [150]. Finally, linear methods compute the shape solution based on the linearisation of the reflectance function in order to obtain either a closed form solution for the depth at each point [154], or to obtain an expression that can be solved iteratively for the shape [155].

In Section 7.2 we present an example of a linear shape from shading algorithm, specifically that of Tsai and Shah [155], for which code is readily available, as a first investigation into the abilities of a general SFS method. We also look at another, closely related technique known as photometric stereo, which is essentially the same technique as SFS in that it uses a simple reflectance model to compute shape from shading information present in an image, but instead of using a single image it uses a minimum of two or three images taken under different light source directions in order to provide the three necessary equations to solve for the three unknowns when attempting to estimate the surface normals, or two equations for the two unknowns when solving for

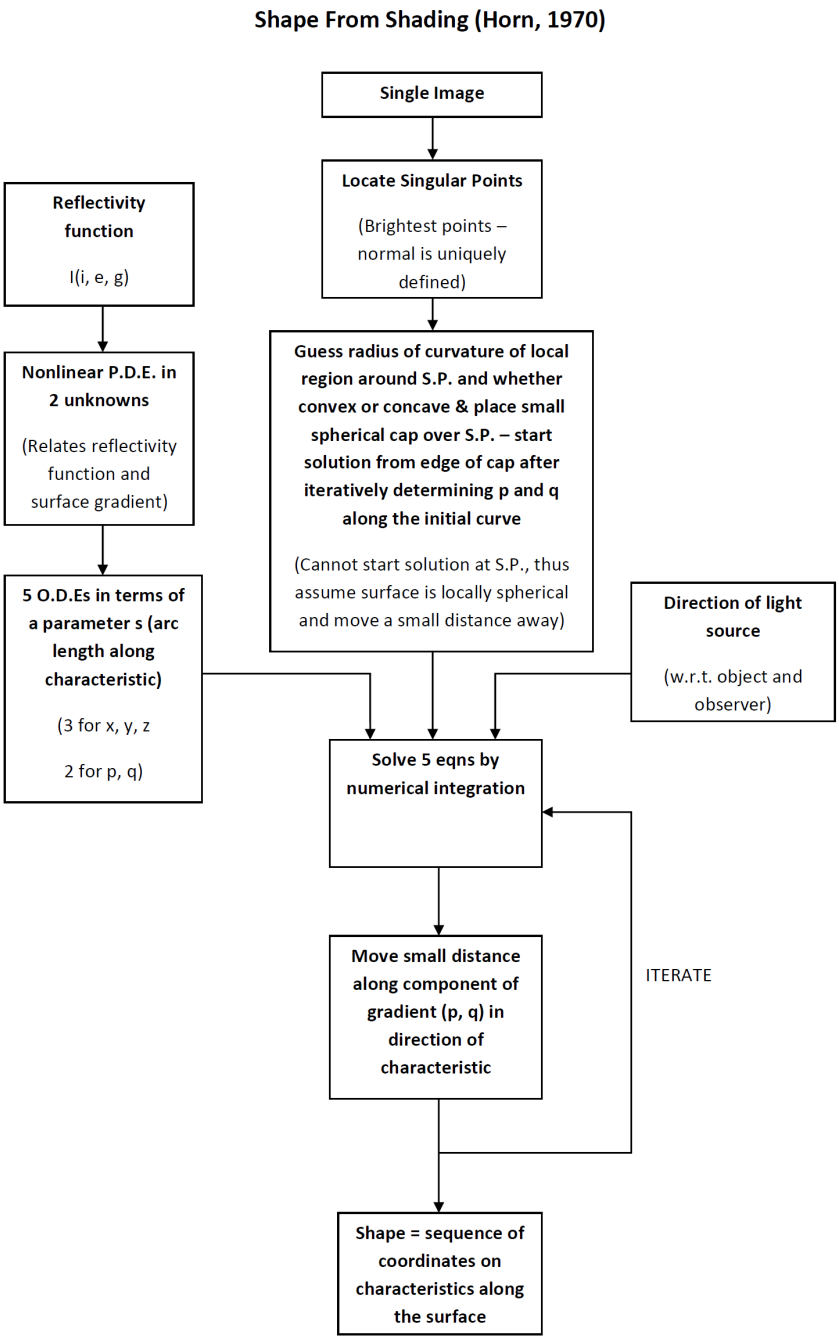


Figure 7.1: Algorithm flow diagram for the original SFS algorithm proposed by Horn [152]

surface gradient. This eliminates the need for additional constraints and assumptions that are imposed on the shape estimates when using only a single image. Again, the method that has been implemented is a general photometric stereo technique, based on the work of Woodham [156], and is mainly used as an examination of the potential of

this type of technique, at this stage. Figure 7.2 presents an overview of the method employed to construct the surface shape using this algorithm. For further details, see [156]. The code for the two algorithms investigated is freely available on the internet, therefore we will not present further details of the algorithms here, if further information is required, please see [155, 156]. In this work these two simple algorithms were tested using PANGU images that are representative of the type of images that may be recorded by the camera during descent, in order to evaluate the applicability of the SFS techniques to the problem investigated in this project.

The ultimate aim would be to combine a shape from shading technique with the structure from motion techniques discussed in the previous chapters. SFS has the potential for constructing a dense DEM of the entire landing site, but is unable to provide absolute depths, therefore combining it with the previous methods should prove complimentary since SFS could fill in the gaps in the DEM from SFM and the SFM algorithm could provide absolute depth information, which would enable a much richer description of the surface structure than either technique alone.

7.2 Preliminary Results

Figure 7.3 presents the results from the Tsai and Shah [155] linear SFS method. Here we have used a single PANGU image from a height of 1000m above the landing site, in which the light source direction is 55° azimuth and 25° elevation. The input image is shown in Figure 7.4.

It can be seen from Figure 7.3 that the results are extremely erratic, and completely inadequate for the intended application. However, this particular method is designed for use on smoothly varying, simple surfaces, with no changes in albedo due to surface blemishes, shadows etc, such as would be present in real images of a planetary surface due to boulders, craters, differing surface material types/rock types. Therefore, it is not surprising that the performance is poor from such a simple algorithm. The purpose of using this particular technique was mainly to get a feel for how SFS algorithms work in general and so it was not a genuine attempt at producing realistic and useful results at this stage. Clearly a much more modern and sophisticated algorithm is required before a true assessment of the capability and applicability of SFS can be carried out.

Figure 7.5 presents the results from the photometric stereo algorithm. In this case we use three images, of the same synthetic planetary surface shown in Figure 7.4, but with three different light source directions: 55° azimuth and 25° elevation, 115° azimuth and 25° elevation, and 270° azimuth and 25° elevation.

Here we can see that the reconstructed surface is less affected by noise than it was for the previous method, which is most likely due to fewer assumptions and constraints being imposed in order to obtain a solution, and due to having more shading information available because of the use of three different images. However, the surface does not appear to vary in height significantly over the entire surface so it appears that the reconstructed surface (although not transformed to absolute height) is too inaccurate to be of use. There are also still a number of spurious depth discontinuities caused by the presence of craters and the shadows cast by rocks on the surface. This is not surprising,

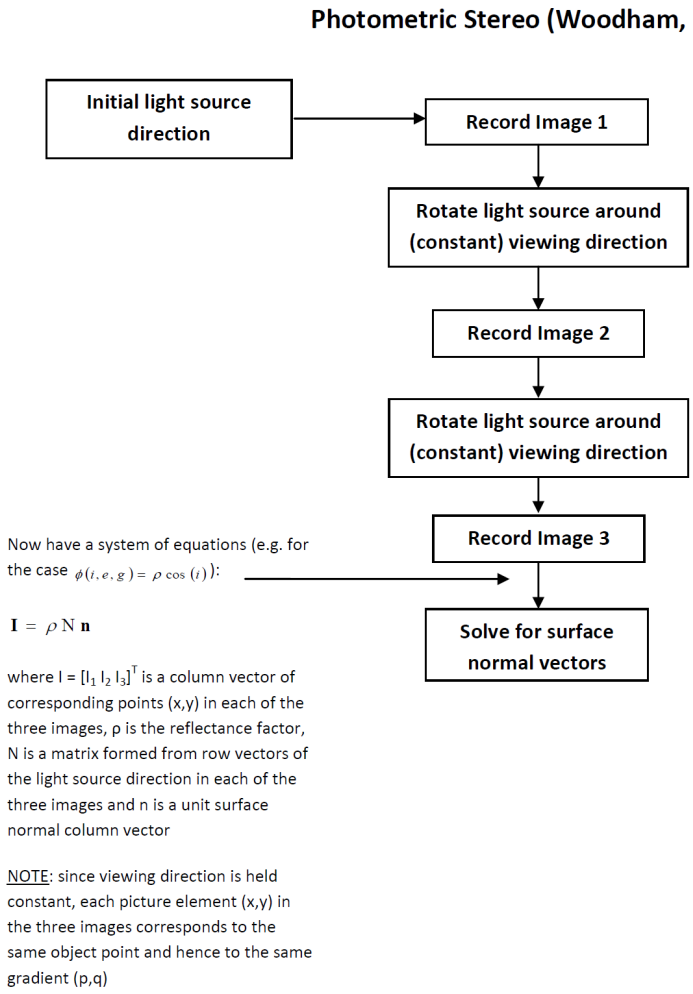


Figure 7.2: Algorithm flow diagram for the original photometric stereo method proposed by Woodham REF HERE!

because it is at these locations where the simple reflectance model will not apply. It should also be said that photometric stereo is unsuited to the application considered in this project as it requires 3 images with the same viewpoint but with different light source directions which could not be obtained during the short period of time of descent for which the spacecraft has significant velocity and control of the light source (the sun) direction is impossible. However, it provides an illustrative example of how improvements can be made when the number of restrictive assumptions can be reduced.

7.3 Conclusions from Simple SFS Approaches

The results presented above would be inadequate for use in the application intended for this project. However, this was to be expected since it was known that the simple

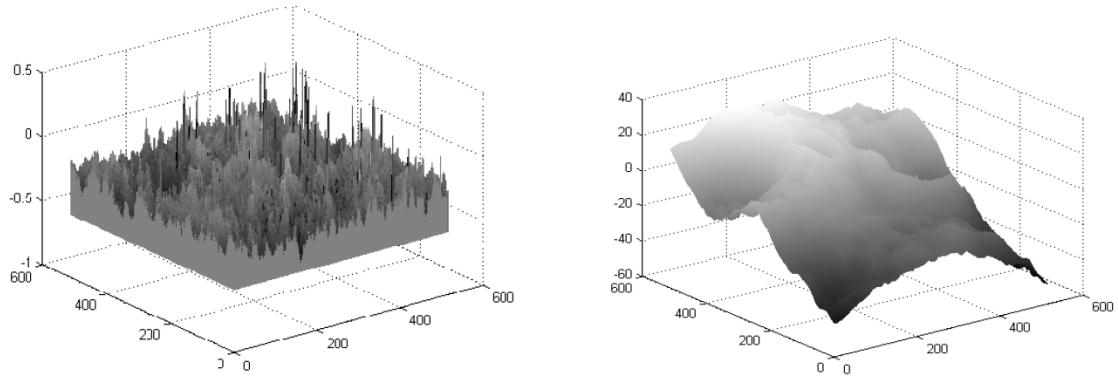


Figure 7.3: Left: Results from Tsai and Shah SFS method (x and y axes are in pixels, z axis represents relative height); Right: Ground truth DEM (x and y axes are in pixels, z axis represents absolute elevation in metres).

Figure 7.4: Input image for SFS algorithm

methods would not be capable of dealing with complex surfaces such as the terrain on a planetary surface. The purpose was mainly to gain an appreciation of the capabilities of the basic methods and gain an appreciation of how much extra sophistication would be required to apply this class of techniques to the case of planetary descent above a complex surface that does not necessarily vary smoothly in shape and may also have a non uniform albedo and/or the presence of shadows. The quite extremely poor results obtained by no means rule out the use of these types of technique, especially since it was expected that the simple techniques would be insufficient at the start, but it is now abundantly clear that a much more in depth investigation into methods with a much

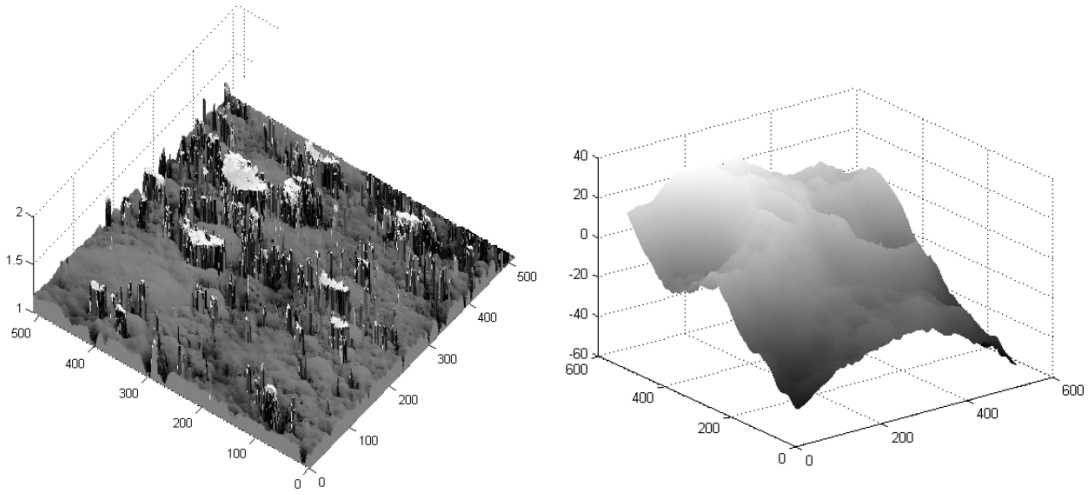


Figure 7.5: Left – Results from photometric stereo algorithm; Right – Ground truth DEM.

greater degree of sophistication is required before they can be applied reliably to the intended application area.

7.4 Slope from Shading

Before moving on to more sophisticated methods of recovering shape from shading information, it was decided to investigate whether there still may be some use for the simple methods if the goal is not the recovery of overall shape, but instead to simply estimate the slope of the terrain directly and determine whether the terrain is too steep. After all, the primary aim for producing DEMs of the landing site is identify the presence of steep slopes that may be too hazardous for the spacecraft to land on. Here, we develop a simple method of estimating the slope of the terrain and mark out whether the terrain is safe or not by comparing the results against a threshold value. Safe areas will be determined on a pixel-by-pixel basis and marked out in green, whereas unsafe areas will be marked out in red.

In order to achieve this, the following assumptions are made:

- Lambertian reflectance model: the surface scatters incident light isotropically. Then, grey levels do not depend on the position of the observer (no privileged direction), but depend only on the angle of incidence, θ;
- Constant albedo on the ground (homogeneous surface, including boulders) – may be too simplistic for real world applications but should be sufficient for PANGU;
- The Sun is a punctual source located at infinity (i.e. light rays are parallel);
- The Sun elevation is known.

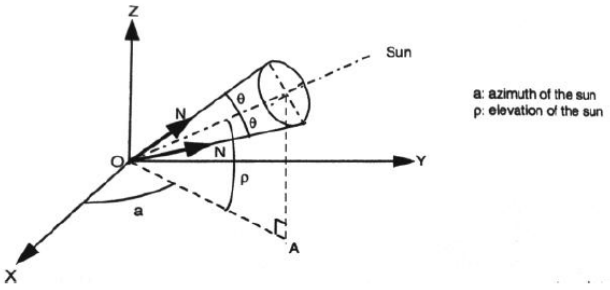


Figure 7.6: Slope from shading geometry

Let θ be the angle between the light source direction and the local surface normal, as shown in Figure 7.6. Under the Lambertian assumption, the reflected luminance in any direction is given by

$$I = aL\cos(\theta) \tag{7.1}$$

where a is the albedo of the surface and L is the illumination received by the surface. Then if we know the relationship between the received luminance and the grey level, G, in the image, we can retrieve $\cos(\theta)$ directly from the grey level.

Assuming that the camera is calibrated, such that when a light ray hits the surface parallel to the surface normal vector, the gray level equals 255, and also assuming that the relationship between the luminance and the grey level is linear (these assumptions are true for PANGU images). Then the relation between the grey level and θ is simply

$$G = 255\cos(\theta) \tag{7.2}$$

Therefore, the solution (θ) does not depend directly on any parameters that relate to the surface properties, i.e. albedo. Thus, it may be more reliable than the previous approaches when used on complex surfaces.

It is important to note that the angle θ does not uniquely define the surface normal vector. It is only possible to know a range of values within which the surface normal vector lies, i.e. for a given θ and light source elevation angle ρ , the surface normal vector belongs to a cone, as shown in Figure 7.6.

The slope angle S (angle from the normal of the horizontal plane and the normal to the surface) then belongs to the interval $[S_{min}, S_{max}]$ where $S_{min} = \pi/2 - \rho - \theta$ and $S_{max} = \pi/2 - \rho + \theta$. It was found through trial and error that the angle S_{min} more accurately represents the surface slope in comparison to the ground truth. The results of this technique are presented in the following section.

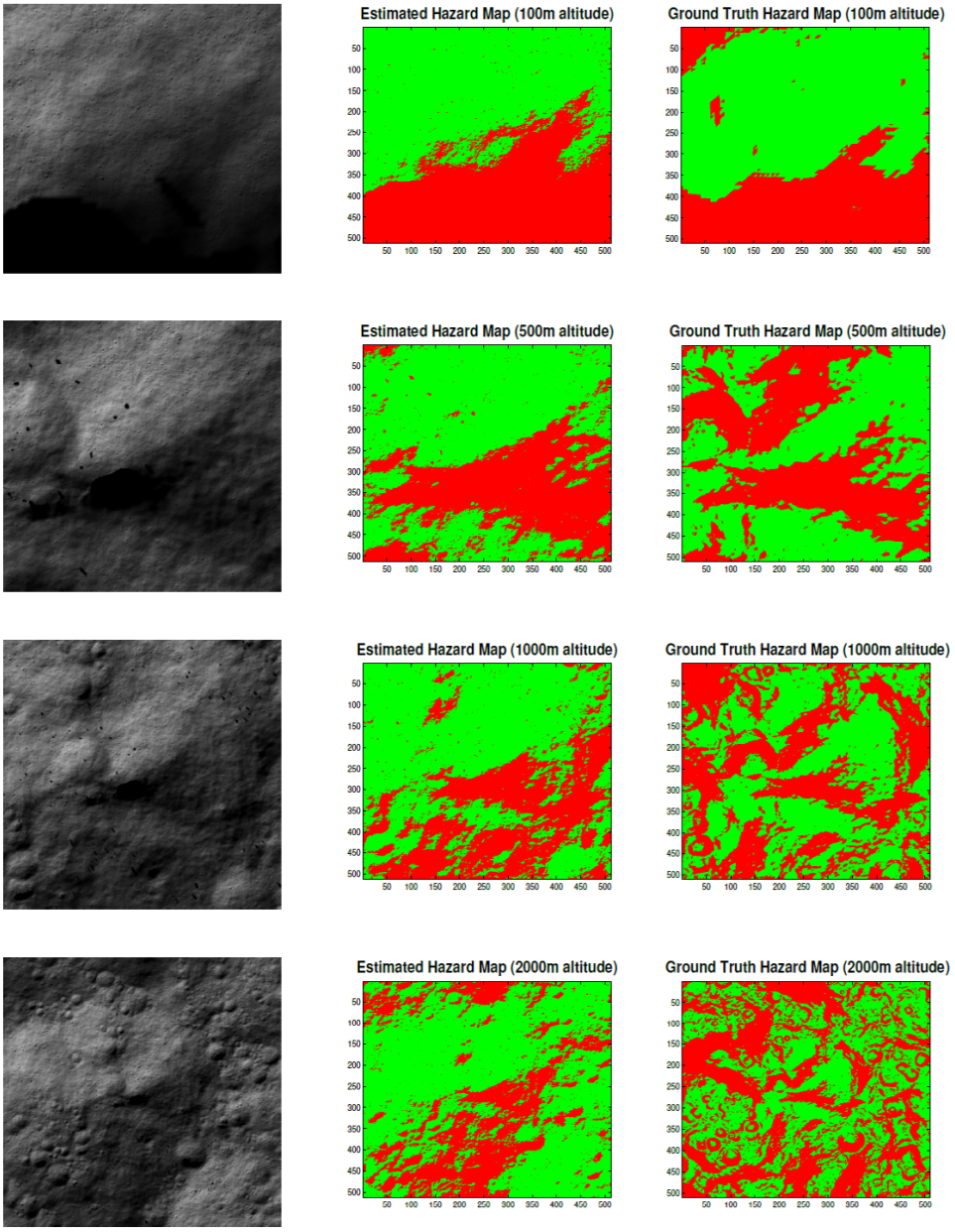


Figure 7.7: Result from the slope from shading algorithm for 4 different test altitudes: 100m (top row), 500m (2nd row), 1000m (3rd row), 2000m (bottom row).

7.5 Slope from Shading Results

Figure 7.7 shows the results obtained from this simple slope from shading method for four different heights ranging from 100m to 2000m above the landing site. Row one of the figure shows the results for the 100m altitude case, with the input image on the left (for which the

light source direction is $55°$ azimuth and $25°$ elevation), the estimated hazard map is in the centre and the ground truth hazard map on the right. There appears to be a very close agreement between the two hazard maps, with most of the hazardous areas captured by the algorithm. There are a few anomalies, where the algorithm appears to have failed to identify hazardous slopes, such as the top left corner, as well as a few points that have been falsely identified as hazards. However, on the whole the performance is quite promising at this altitude. As we go on to higher altitudes it appears that the performance decreases, although there are still areas that are in close agreement with the ground truth. However, the number of missed hazardous areas increases, as does the number of false positives. This can be attributed to there being more complex terrain features visible in the higher altitude images, such as craters with shadowed areas, etc. In particular, the algorithm appears to perform well on the shadowed areas, but on the other hand it fails to identify steep slopes on the opposite side of the craters (the unshadowed side). This type of problem can be seen quite clearly on the bottom row of the figure for the 2000m altitude case, which also shows many other areas that the algorithm has failed to identify. Craters aside, the algorithm in general appears to perform better on the darker regions that it does the lighter regions. Overall, however, there appears to be some promise to this approach and so it may benefit from further development.

7.6 Slope from Shading Conclusions

A simple slope from shading algorithm has been developed as a means of trying to avoid the problems that plagued the direct shape from shading algorithms when they were used with the complex surfaces that are necessary in this work. The aim was to directly estimate the slope of the terrain using a simple technique, instead of using shading information to produce a DEM of the landing site that could then be used to identify hazardous slopes. This approach appeared to yield more reliable results since it does not directly depend on parameters relating to the properties of the surface. Overall the results were promising, but a number of regions were misidentified, especially at higher altitudes. This simple technique shows a reasonable amount of promise, however it is doubtful that it would be reliable enough to guarantee a safe landing, as it can quite easily miss hazardous areas. For this reason it was decided to move on to investigating more sophisticated, modern shape from shading techniques in an attempt to obtain a solid foundation upon which we could incorporate the SFM results to produce a combined approach that should lead to improved accuracy and a full scale DEM.

7.7 Modern Shape From Shading Approaches

The SFS survey paper by Zhang et al [150], presents a detailed review of a large number of techniques that have been applied to the problem of recovering shape from shading information, and even includes a comparison of six different methods in an attempt to determine their relative performance and suitability to specific types of applications. These methods can, however, be regarded as classical methods since they are primarily applied to the recovery of simple, smoothly varying objects and were

predominantly applied to synthetic, monochrome images or very simple real images. No attempt was made to apply the discussed methods to complex shapes such as planetary surfaces that may contain shape discontinuities, shadows and widely varying depths. A more up to date survey of the field of shape from shading was presented by Derou et al [157], in which a slightly different classification of the SFS methods was given – instead of the four classes of technique given in [150], the authors of [157] classified the techniques in to 3 categories: methods based on partial differential equations, methods using optimisation, and methods approximating the image irradiance equation. Three techniques, one from each class, were implemented and a thorough comparison was made between the 3 approaches in order to determine the efficiency and accuracy of the methods when applied to synthetic and real images. Unfortunately, these methods were again only tested on images of very simple, smoothly varying shapes, so it is difficult to conclude their applicability to a more complex problem such as that tackled in this project.

The primary problem in the reviewed shape from shading techniques that make them difficult to apply to natural images and more complex surfaces, aside from the problem being under-determined, which is inescapable, is that albedo is assumed to be uniform across the entire surface and almost always is assumed to be known *a priori*. Therefore, rather than potentially endlessly experimenting with the various techniques reported in the literature that have been tested on images of comparatively simple objects, we chose to look for methods that have been specifically designed to cope with scenarios that are similar to our own and fortunately there are a few examples.

One of the first examples of using shape from shading in a related scenario to that of ours (other than in the somewhat related field of photoclinometry) can be found in the work of Thompson [158], which describes a technique for determining surface topography from orbital images by combining shape from shading and stereo vision. However, this method requires two images from different locations (sufficient to give a suitably large baseline separation) and with different lighting conditions, therefore this approach would not be suitable for use in planetary descent where the lighting conditions would be expected to remain constant over the descent image sequence. Along a similar line as the work of Thompson is the work of Heipke et al [159–163], which is a method of combining image matching, between (at least) two separated views of the scene, and shape from shading. This method was named multi-image shape from shading and claims to fully combine a global least-squares based image matching algorithm with a simple shape from shading formulation. A consequence of this is that the two methods compliment each other, with the image matching method being able to determine the full-scale depths of highly textured regions in the image and the SFS part being able to fill in the gaps in low textured regions, and overall the combined method is able to produce more accurate results than either of the two methods applied in isolation [159]. Originally, this method required a constant albedo across the entire surface but later it was extended to enable the albedo to vary over the surface in a piecewise manner. The algorithm was tested using synthetic images [159, 160], real aerial images [161], and, most importantly for this work, with real images of the surface of the Moon recorded by the NASA Clementine spacecraft [162, 163], and was found to produce accurate results in comparison with ground truth DEMs produced using

photogrammetric stereo techniques.

Another method that focussed on 3D surface reconstruction of part of the Lunar surface is that of Wohler [164], which presented two slightly different methods of reconstructing the surface. The first technique was based on a single image shape from shading scheme, but used two further images of the scene that contained shadowed regions, which were used to initialise the main shape from shading algorithm. The second method tackled the reconstruction problem by the use of a quotient based intensity error function formed from the intensity values of two separate images of the same scene under different lighting conditions. The first technique, as with the majority of shape from shading schemes, requires an assumption of uniform albedo over the surface, where as the second method can handle cases in which the albedo is variable. Both methods, however, require a minimum of 3 or 2 (respectively) images with exact pixel-to-pixel correspondence and each image is required to have different light source orientations, therefore both methods can be regarded as photometric stereo methods rather than strict shape from shading algorithms, and therefore would not be of much use in our work. Another interesting method involving Wohler is presented in Herbort et al [165], in which a method of combining shape from shading and active range scanning is described. The motivation for this method was stated as a complimentary combination of SFS, which yields dense surface detail on small scales, and active range scanning, which can be noisy at fine scales but reliable on large scales. The same can be said for our aim of merging a feature based SFM method and shape from shading. Unfortunately, the method presented in [165] is of limited use in our case since we cannot assume the presence of an active scanning range sensor such as laser altimeters/LIDAR as was used in the testing of their method.

O'Hara and Barnes [166] presented an optimisation based SFS algorithm that was tested on images captured by the Mars Express HRSC instrument and was found to provide improved fine surface detail over that derived from stereo methods. This algorithm was based on a different way of representing depth compared to other SFS methods. Instead of a discrete height grid, where each discrete point of the grid represents a height, a depth grid is used where each point represents a depth along unit viewing directions, thus the resulting depth map is related to a range image for the perspective camera model (i.e. where the range is the hypotenuse of the triangle formed from the origin of the camera frame, the Z axis and the point (X, Y, Z)). Note that unit vectors are used, however, because the scale is unknown from a single image. The method allows for different types of camera models to be used, such as orthographic (as is typically employed in SFS), perspective, and push-broom, and was implemented using two different types of reflectance model: the standard Lambertian model, and the Oren-Nayar diffuse reflectance model. However, the method requires known and uniform albedo, which is an often needed requirement in SFS, and needs to be used with images that have been pre-calibrated so that image intensity is proportional to image irradiance, which is also not uncommon in SFS.

Finally, a sophisticated SFM method that was applied to images of the Lunar surface captured during the Apollo missions was developed by Barron and Malik and presented in [167]. This method stands out as it is possibly one of the only methods that truly allows for variable unknown albedo. The algorithm is based on an optimisation approach

to SFM, in which the optimisation is carried out in a coarse-to-fine iterative refinement scheme using Gaussian and Laplacian pyramids, which, respectively, represent the albedo and depth estimates that together describe the most likely albedo and shape that explains a single image. Thus, in this algorithm both shape and albedo are estimated simultaneously. The pyramid representation of scene depth naturally lends itself to the (optional) incorporation of an initial, low-frequency depth estimate (used as the top-most (coarsest) level of the pyramid), such as that which may have been derived from texture based stereo vision measurements of the depth of the scene, or in our case from the sparse DEMs produced using feature based SFM. Therefore this method is able to optimally combine the strengths of stereo or SFM methods, which are most suited to coarse scale depth measurements from high textured image regions, with SFS, which is best suited to low textured image regions and the estimation of depth changes at fine scales. This method was later further developed to also enable simultaneous estimation of the illumination direction along with albedo and shape in a technique now known as Shape Illumination and Reflectance From Shading (SIRFS) [168]. The flexibility of this method, particularly in that almost no prior knowledge about the surface properties (albedo) and light source are required, and that it very easily allows for the incorporation of external depth information from SFM, are the reasons why this method was chosen for further investigation for our application. Further details about this method will now be presented in Section 7.8.

7.8 Shape, Illumination, and Reflectance From Shading

A long-time problem in shape from shading algorithms, that limits their applicability to real-world imagery, especially those that can be regarded as images of natural scenes, is that albedo is generally assumed to be both uniform and fully known. Uniform albedo will generally only be true for simple man-made objects and possessing full albedo knowledge will rarely occur in practice when the objects in the scene are not easily accessible or if it is impractical to perform a detailed examination of the reflective properties of the materials of all the objects that are to be imaged. These difficulties are particularly pronounced when dealing with the type of natural scenes produced by the terrain of planetary surfaces that have not previously been visited and therefore it is unknown exactly what will be encountered. The Shape and Albedo From Shading algorithm (SAFS) developed by Barron and Malik [167] (later extended to Shape, Illumination and Reflectance From Shading (SIRFS) [168]) is, to this authors knowledge, the first SFS algorithm to truly relax these assumptions and allow for direct estimation of unknown albedo that is variable across the surface of the imaged object. This is achieved by imposing “naturalness” priors over the albedo and shape in order to simultaneously recover the most likely albedo and shape that explain a single image [167]. These naturalness priors are applied over multiple scales via a Laplacian image pyramid that represents a coarse-to-fine iterative refinement estimation strategy for recovering the most likely albedo map of the scene, and similarly over a Gaussian pyramid representing coarse-to-fine depth estimation in the scene. The use of image pyramids was reported in [167] to dramatically improve performance and has an extra

benefit of quite elegantly enabling a coarse initial estimate of depth to be incorporated, such as that obtained from a stereo vision or SFM algorithm, by scaling it down and using it as the top-most (coarsest) level of the Gaussian depth pyramid. The SAFS algorithm then refines this initial estimate by effectively filling in the gaps between the high textured regions using the relative differences in shading in the low textured areas to produce a much finer scale DEM of the scene.

The method for recovering shape from shading in [167] produces a full depth map, $\hat{Z}$, and albedo map, $\hat{\rho}$, from an input image, I, the light direction, L, and an optional coarse depth, Z_0, via the minimisation of the following composite cost function:

$$f^{safs}\left(Z_{\mathcal{L}}\right) = f^{\rho}\left(\mathcal{G}\left(\rho\left(\mathcal{L}^{-1}\left(Z_{\mathcal{L}}\right)\right)\right)\right) + f^{Z}\left(Z_{\mathcal{L}}\right) \tag{7.3}$$

where $f^{\rho}(\cdot)$ is the contribution to the cost due to albedo estimation, $f^{Z}(\cdot)$ is the contribution to the cost due to depth estimation, $\mathcal{G}(\cdot)$ represents an operation that constructs a Gaussian pyramid from an image (an albedo image in this case), $\mathcal{L}(\cdot)$ is an operation that constructs a Laplacian pyramid from an image (a depth image in this case), and therefore $\mathcal{L}^{-1}$ reconstructs the image from a Laplacian pyramid, $\rho(Z)$ is the albedo image, which is a function of the depth-map Z, and $Z_{\mathcal{L}}$ is the Laplacian pyramid representation of the depth-map of the scene. The output of the algorithm is obtained via a non-linear conjugate gradient descent algorithm to minimise Equation (7.3) and produce the most likely estimate of Z:

$$\hat{Z} = \mathcal{L}^{-1}\left(\arg\min_{Z_{\mathcal{L}}} f^{safs}\left(Z_{\mathcal{L}}\right)\right), \qquad \hat{\rho} = \rho(\hat{Z}) = \frac{I_{x,y}}{S_{x,y}(\hat{Z})} \tag{7.4}$$

where $I_{x,y}$ is the input image and $S_{x,y}(\hat{Z})$ is a Lambertian rendering based on the estimated depth $\hat{Z}$ under light source L.

The Lambertian rendering of the estimated depth is carried out by assuming Lambertian reflectance and orthographic projection, which was stated in [167] to improve performance compared to other projection and reflectance models. Specifically, each pixel $(x,\, y)$ is considered to be bounded by four points whose depths are calculated using bilinear interpolation, and then rendered based upon an average of the inner-products of the light source direction vector and the two unit normal vectors of the two triangles formed by the 4 points. The unit normals are calculated from the following expressions:

$$\begin{aligned}
\mathbf{n}^{+}_{x,y}(\hat{Z}) &\propto \begin{bmatrix} \hat{Z}(x-1/2,\, y-1/2) - \hat{Z}(x+1/2,\, y-1/2) \\ \hat{Z}(x-1/2,\, y-1/2) - \hat{Z}(x-1/2,\, y+1/2) \\ 1 \end{bmatrix} \\
&, \\
\mathbf{n}^{-}_{x,y}(\hat{Z}) &\propto \begin{bmatrix} \hat{Z}(x-1/2,\, y+1/2) - \hat{Z}(x+1/2,\, y+1/2) \\ \hat{Z}(x+1/2,\, y-1/2) - \hat{Z}(x+1/2,\, y+1/2) \\ 1 \end{bmatrix}
\end{aligned} \tag{7.5}$$

then $S_{x,y}(\hat{Z})$ is computed as follows using the Lambertian reflectance model:

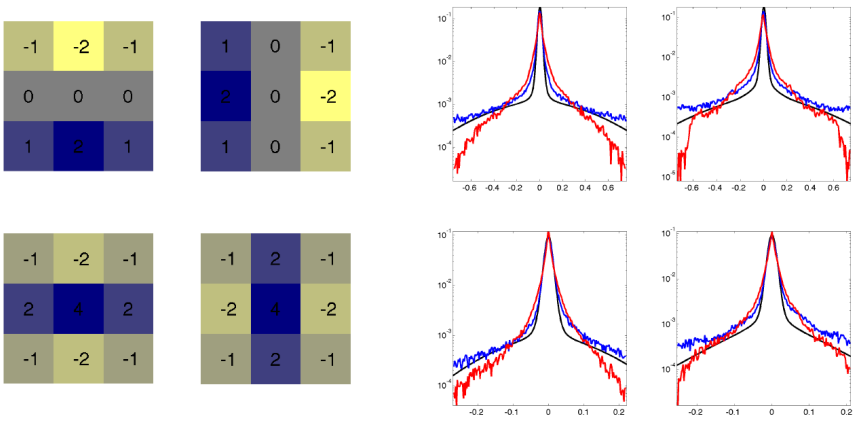
$$S_{x,y}(\hat{Z}) = \frac{1}{2}\left(\max\left(0, L \cdot \mathbf{n}^{+}_{x,y}(\hat{Z})\right) + \max\left(0, L \cdot \mathbf{n}^{-}_{x,y}(\hat{Z})\right)\right). \tag{7.6}$$

The strategy for estimating albedo involves convolving each level of the estimated Gaussian albedo image pyramid with a set of 4 oriented edge and bar filters of the form shown in Figure 7.8 (a). The responses to these filters are very different on pure albedo images than they are with natural images (i.e. shading $\times$ albedo), which can be seen in Figure 7.8 (b) in which the log-histograms of the responses to the filter bank are presented and where it can be seen that the responses of albedo images are much more kurtotic in comparison with natural images. It is this difference that enables the albedo of the surface to be separated from the shading due to depth in natural images. To make use of this distinguishing property of albedo images and derive the cost function for albedo, f^{ρ}, a "Field of Experts" (FoE) type model is utilised and applied to the albedo Gaussian pyramid. The FoE model was developed by Roth and Black [169] as a method of learning generic global image priors that accurately capture the statistics of natural images through a combination of the Products of Experts (PoE) framework (for modeling the complex statistics of small image patches by taking the product of several simpler expert distributions) and Markov Random Field (MRF) models (which establishes a non-causal statistical relationship between a pixel and its immediate neighbours). The FoE method was applied to image denoising and image inpainting in [169], and to optical flow in [170], and it was found to outperform certain specialised techniques even when trained on generic image datasets and not tuned towards a specific application [169].

Traditionally, MRF methods utilise a simple neighbourhood system, such as a first order, 4-neighbour system, or a second order, 8 neighbour system, as shown in Figure 7.9 (a), and where the statistical dependence between pixels is assumed to extend only to the most immediate neighbouring pixels. That is, given a 2D, $N \times M$ rectangular image lattice $\Omega = \{(i,j) \mid 0 \leq i \leq N-1,\ 0 \leq j \leq M-1\}$, where each pixel $s \in \Omega$ is associated with a random variable X_s representing the set of possible intensity values (or some other label in image segmentation, for example), a neighbourhood system η consisting of a set of neighbourhoods $\eta_s \subset \Omega$, where $s \notin \eta_s$ (i.e. the pixel s is not part of its own neighbourhood) and $s \in \eta_t \rightarrow t \in \eta_s$ (i.e. s is a part of some other pixel, t's neighbourhood, where t is a neighbour of s). Then, the statistical dependence between a pixel, s and its neighbours is determined by defining a clique potential $V_c(x)$ on each clique c, taking the exponential of the sum of the clique potentials of all cliques C in Ω, and dividing the exponential by a normalising constant Z, i.e.

$$P(x) = \frac{1}{Z}\exp\left\{\sum_{c \in C} V_c(x)\right\}, \tag{7.7}$$

to form a Gibbs Random Field (GRF), which by the Hammersley-Clifford Theorem is equivalent to the MRF. As an example, for an 8-neighbour system, the types of cliques are shown in Figure 7.9 (b), where a clique is a set of pixels (including a single pixel), such



(a) Filter bank (b) Filter responses

Figure 7.8: (a) Filter bank, and (b) log-histogram filter responses, used in estimation of albedo over a Gaussian pyramid of albedo images. Note that in (b) the responses to the filter bank are much more kurtotic for albedo images (blue) than they are for natural images (red). The black curve is a Gaussian Scale Mixture model for albedo learnt from a set of training images. (Image source [167])

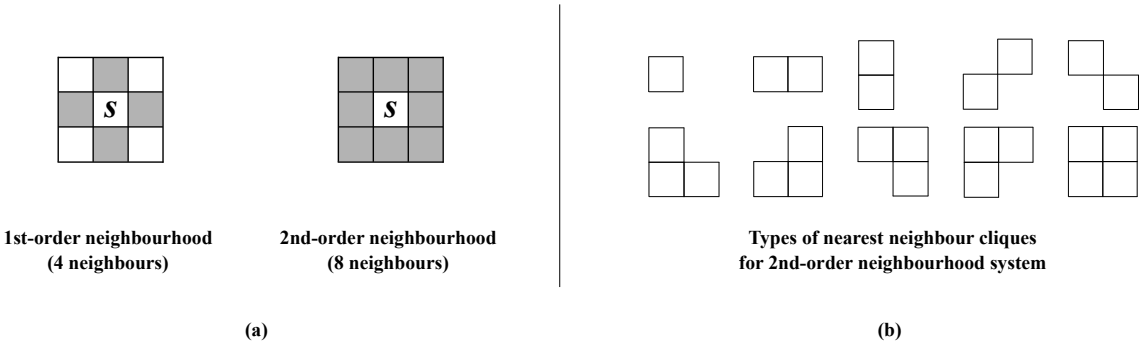


Figure 7.9: (a) Two examples of neighbourhood systems - 1st order and 2nd order neighbourhoods; (b) Nearest neighbour cliques for the 2nd-order neighbourhood system.

that any two elements in the clique are neighbours of each other. The neighbourhood system employed in [167] is the 2nd-order neighbourhood system, but as in the FoE model [169,170], a higher order MRF is used in order to extend the statistical dependency between pixels beyond simple nearest neighbour pairs to a 3x3 maximal clique system, i.e. all pixels in a 3x3 pixel grid are considered to neighbours of each other. This 3x3 clique system is shown in Figure 7.10 in which the links (graph edges) between the pixels are represented by the various coloured lines.

As in the FoE model used in [171], the clique potentials are modelled on Gaussian Scale Mixtures (GSM) and applied only to the maximal 3x3 cliques. The GSM potentials for each of the for filters are learned from a training set of albedo images using expectation maximisation [167]. However, unlike other FoE models, the model in [167] is applied to a

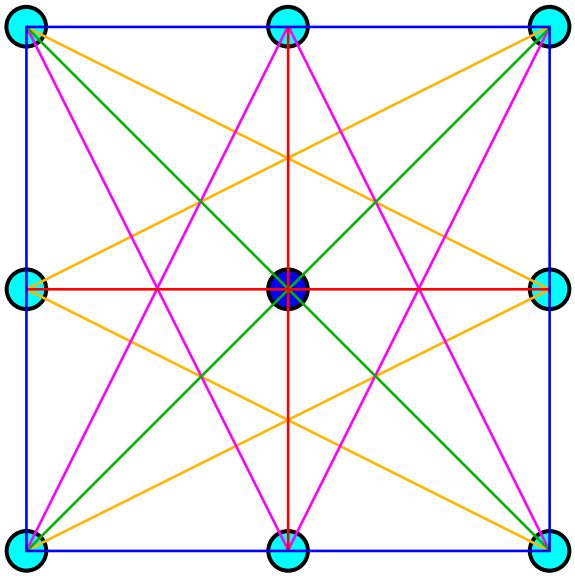


Figure 7.10: 3x3 maximal clique used in FoE type models

Gaussian pyramid rather than a single image, which is equivalent to utilising a multi-scale filter. The reason for using a pyramid representation of the FoE model is because errors in the depth map at low frequencies (higher/coarser levels of the pyramid) can corrupt the implied albedo map at low frequency scales, whereas in the usual FoE models applied to such tasks as image de-noising, the image is only corrupted at the finest scale [167]. The cost function for f^{ρ} is obtained by applying the negative log-likelihood to the multi-scale FoE model, which results in the following expression:

$$f^{\rho}(\rho_{\mathcal{G}}) = -\sum_{k=1}^{K} \lambda_k^{\rho} \sum_{c \in C_k^{\rho}} \sum_{i=1}^{4} \log \left(\sum_{j=1}^{M} \alpha_{i,j,k} \cdot \mathcal{N}\left(\mathbf{J}_i^T \mathbf{x}_{(c)};\ \mu_{i,k},\ \sigma_{i,j,k}^2\right) \right) \tag{7.8}$$

where $\rho_{\mathcal{G}} = \mathcal{G}(\rho(\mathcal{L}^{-1}(Z_{\mathcal{L}})))$ is the Gaussian pyramid representation of albedo implied by $Z_{\mathcal{L}}$, K is the number of levels of the Gaussian pyramid, C_k^{ρ} is the set of all 3x3 maximal cliques in Ω for the k^{th} level of $\rho_{\mathcal{G}}$, $\mathbf{x}_{(c)}$ is the 3x3 patch in $\rho_{\mathcal{G}}$ corresponding to clique c, $\mathbf{J}_i$ is the i^{th} filter, of which there are 4 (see Figure 7.8), $\alpha_{i,j,k}$ are the mixing weights of Gaussian ik of the GSM's, of which there are $M = 50$, and each of which has variance $\sigma_{i,j,k}^2$ and all of which have mean $\mu_{i,k}$, and finally λ_k^{ρ} are the hyperparameters that weight each scale of the prior, which are tuned to maximise SAFS performance on the training set [167].

For the Laplacian pyramid representation of depth, there are 2 goals stated in [167] that assist the construction of priors: (1) the residual low-pass level of the pyramid should remain close to the initial low-pass observation Z^0, and (2) $Z_{\mathcal{L}}$ should be regularised using a statistical model learned from example depth maps. The first goal is satisfied by assuming that Z^0 is a noisy observation of $Z_{\mathcal{L}}[K]$ for which it is assumed that the noise is Gaussian and independent and identically distributed, and the second goal is achieved by maximising the log-likelihood of $Z_{\mathcal{L}}$ under a 4-connected multi-scale MRF, in which the

clique potential is a bivariate GSM. Thus, using the negative log-likelihood of these priors as the cost function $f^Z(Z_\mathcal{L})$, the following expression is obtained:

$$f^Z(Z_\mathcal{L}) = \lambda_K^Z \left\| Z_\mathcal{L}[K] = Z^0 \right\|_2^2 - \sum_{k=1}^{K-1} \lambda_k^Z \sum_{c \in C_k} \log \left(\sum_{j=1}^{M} \alpha_{j,k} \cdot \mathcal{N}\left(\mathbf{x}_{(c)};\ \mu_k,\ s_{j,k} \cdot \Sigma_k\right) \right). \tag{7.9}$$

This is similar to Equation (7.8), except that we have $K-1$ bivariate GSMs (each with a single covariance matrix Σ_k), i.e. one single bivariate GSM (for each level) is applied to each maximal clique (in this case given by the nodes connected by the red cross in Figure 7.10), instead of filter banks, and a squared error term against Z^0 at level K. λ_k^Z are the hyperparameters for each level that are tuned to provide maximum performance on the training set.

The algorithm just described was extended upon in [168] to include an extra term in the cost function that represents a prior over illumination. This prior is simply a fit of a multivariate Gaussian to a spherical harmonic representation of the illumination direction:

$$f^L(L) = \lambda_L \left(L - \mu_L\right)^T \Sigma_L^{-1} \left(L - \mu_L\right) \tag{7.10}$$

where μ_L and Σ_L are the parameters of the Gaussian that is to be learnt from the input data, and λ_L is the multiplier on this prior, which is learned from the training data.

The SIRFS algorithm is what we apply to the descent images used in this project, mostly due to the availability of source code. However, the discussion above focussed more explicitly on the SAFS algorithm (which is basically SIRFS without the illumination estimation) because this algorithm was applied to imagery that closely related to that used in this project and is therefore of high relevance.

7.9 Application of SIRFS to Descent Images

In this section we present the results of some initial tests of the SIRFS algorithm on a synthetic image generated from PANGU. This image is the first image in a PANGU sequence starting from 2000m altitude and descending to the surface at a constant pure vertical velocity of 100m/s and with a constant anticlockwise rotation about the Z_w axis of 1rpm. We use this image dataset because we are now moving towards a method of combining the two techniques of SFM and SFS and this is a simple test case that has been used previously in SFM.

Figure 7.11 presents the PANGU image used in the following tests of the SIRFS algorithm along with its ground truth DEM with exact pixel-to-pixel correspondence between the DEM and the input image, viewed from a top-down perspective.

Before proceeding to more challenging tests, we first examine the performance of SIRFS when the actual ground truth DEM is used as the input depth map Z^0, since this should produce the most accurate results, and therefore it gives a baseline performance assessment of the accuracy capabilities of the algorithm. The results of this test are

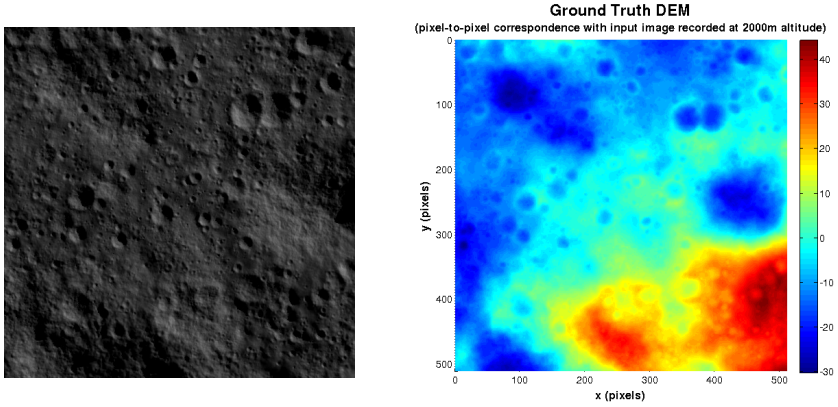


Figure 7.11: PANGU image, at 2000m altitude, used for testing SIRFS (left), and ground truth DEM with exact pixel-to-pixel correspondence with input image (right)

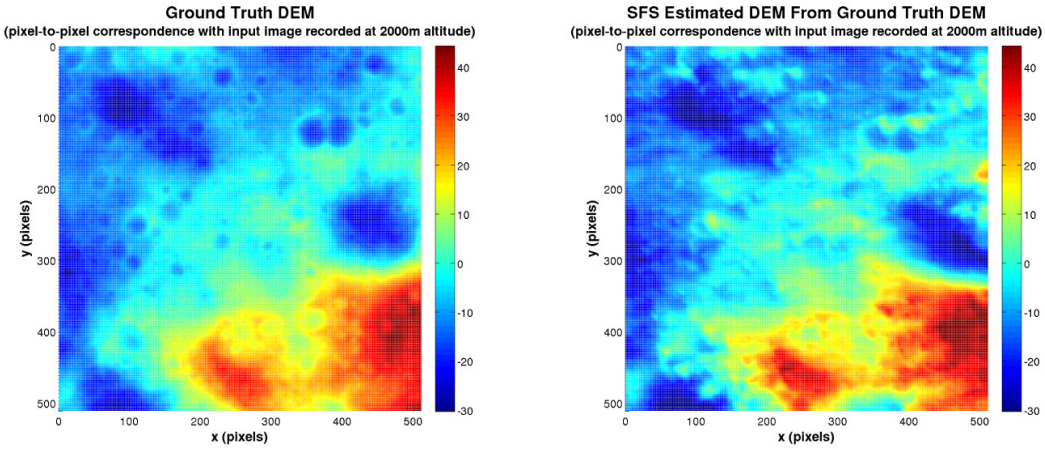


Figure 7.12: SIRFS test using full ground truth DEM as input depth map Z^0

presented in Figure 7.12, from which it can be seen that, in general, the DEM is reproduced in very close agreement to the input ground truth DEM. However, there do appear to be some discrepancies, which are most readily seen in the distorted shapes of the large depression centred around the point with approximate coordinates $(x_w, y_w) = (450, 250)$, and the elevated region shown in red in the bottom right corner region. In general, it appears that the re-produced DEM shows higher-frequency surface elevation variations than the input DEM, which might be an indication of over fitting, that could perhaps be rectified by a different choice of input tuning parameters (see the instructions in the README.txt file in the code download available from https://people.eecs.berkeley.edu/~barron for details on these parameters), however, a more satisfactory set of tuning parameters could not be found in this investigation that provided better results, although the search was by no means exhaustive. The discrepancies between the ground truth DEM can be seen more clearly in the error DEM shown in Figure 7.13, which verifies that over much of the DEM the

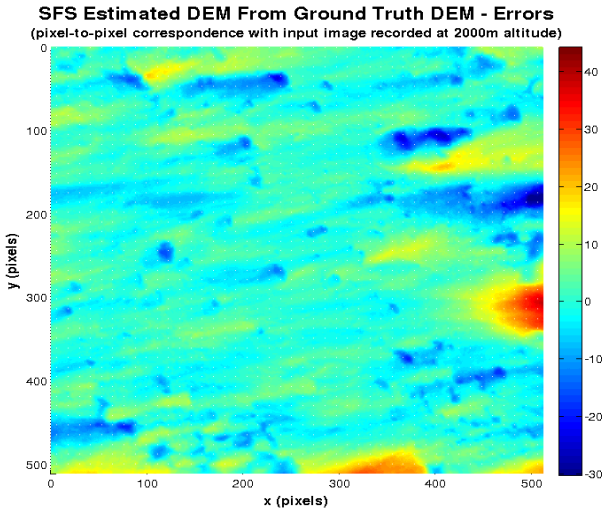


Figure 7.13: Errors in SIRFS results from test using full ground truth DEM

errors are close to zero: approximately 22% of the SIRFS DEM has errors in the range of ± 1m, 40% of the DEM has errors in the range of ± 2m, and for 55% of the DEM the errors are within ± 3m; but there are large discrepancies in the areas mentioned in which the errors can be seen to be as large as ± 30m, and a few other small areas in which the errors are approximately within the range of ± 10 to $\pm\ 20$m. These areas of high error are responsible for driving up the overall RMS error to around 6m.

Since the DEMs produced from SFM are very sparse, an assessment of the performance of SIRFS with differing quantities and quality of input information is carried out below. The SFM DEMs are not only sparse, but they also do not cover the entirety of the terrain area within the field of view of the first image, and so an analysis of how SIRFS behaves with missing information is also required. However, we first investigate the sparsity issue separately by artificially downgrading the density of the ground truth DEM. This is achieved by scaling down the ground truth DEM by constructing an image pyramid and then resizing each resulting DEM and interpolating using bilinear interpolation in order to obtain DEMs of equal size to the original ground truth DEM, but with successively reduced detail. The results for each pyramid level, representing a reduction in detail from $1/2$ down to $1/32$, will now be presented.

Figure 7.14 presents the results obtained from the SIRFS algorithm when the input depth map is derived from the ground truth DEM after it has been reduced in size by a factor of 2 and then resized back to the original DEM size using bilinear interpolation. The difference in quality of the input DEM from the full ground truth DEM is barely noticeable and therefore it is no surprise that the resulting SIRFS estimated DEM appears to be extremely close to the results obtained using the full ground truth as the input depth map. The similarity in outputs from this test and the previous test can be seen in Figure 7.15, which not only shows the DEM errors with respect to the ground truth (left), but also shows the difference between the results of this test and the previous test (right). The error plot on the left of Figure 7.15 is very similar to that presented in Figure 7.13, which is backed

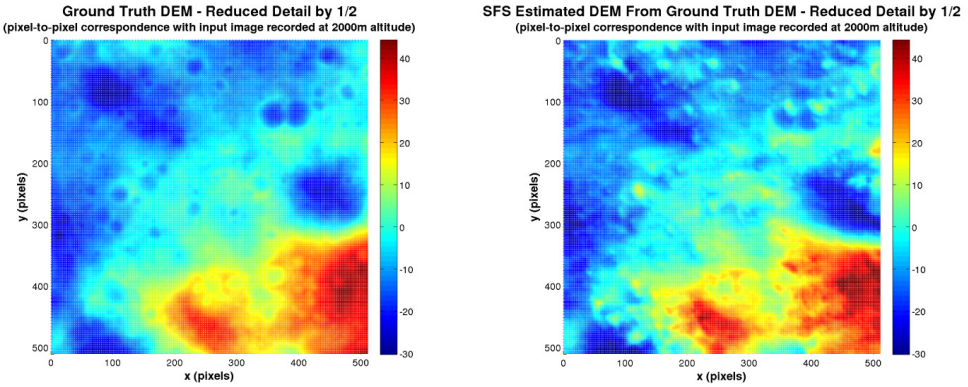


Figure 7.14: SIRFS test using ground truth DEM reduced in detail by $1/2$: Left – input DEM from pyramid level 1 representing a reduction in detail by a factor of $1/2$; Right – SIRFS output DEM

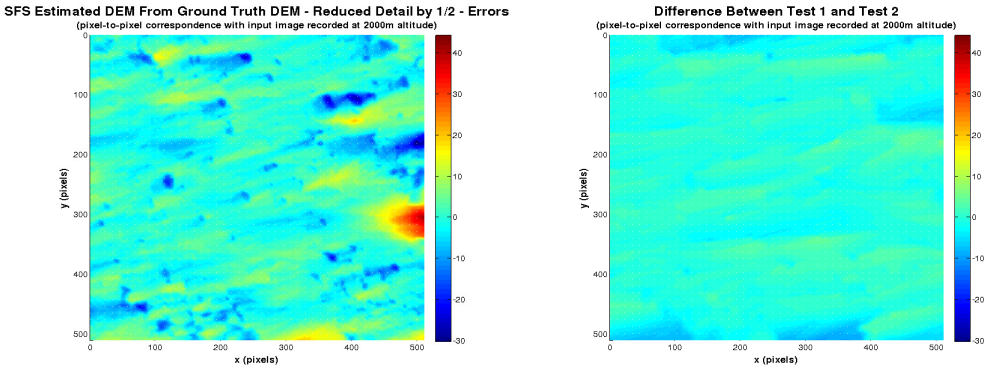


Figure 7.15: Errors in SIRFS results from test using ground truth DEM reduced in quality by $1/2$ (left), and comparison between the SIRFS output from this test and the test on the full ground truth DEM

up by the visualisation of the difference between the output of the SIRFS algorithm for this test and that of the test using the full ground truth DEM as the input depth map, in which the errors are close to zero almost everywhere (fluctuating about zero by around ± 2m), except for a few regions close to the image boundaries where some larger discrepancies of around ± 5m are observed. More specifically, approximately 24% of the SIRFS DEM is in error with respect to the ground truth DEM by ± 1m, 43% of the DEM is in error by ± 2m, 58% of the DEM is in error by ± 3m, and the overall RMS error is approximately 5.4m, which is actually better than the results obtained using the full quality ground truth depth map as the input Z^0, which is a surprising result, but is most likely due to the tuning parameters being closer to optimal values in this case than they were for the previous test. Alternatively, this may point to over-fitting in the full quality ground truth case, which now does not occur due to the reduction in quality of the initial input DEM.

In the next test we use an input DEM derived from the ground truth DEM with a reduction in detail by a factor of 4. The results of this test are shown in Figure 7.16 and again it can seen that the SIRFS output is very similar to that obtained in the previous

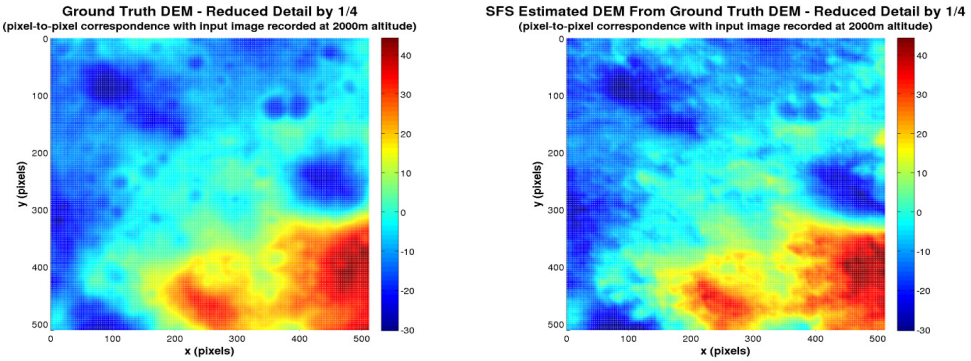


Figure 7.16: SIRFS test using ground truth DEM reduced in detail by $1/4$: Left – input DEM from pyramid level 3 representing a reduction in detail by a factor of $1/4$; Right – SIRFS output DEM

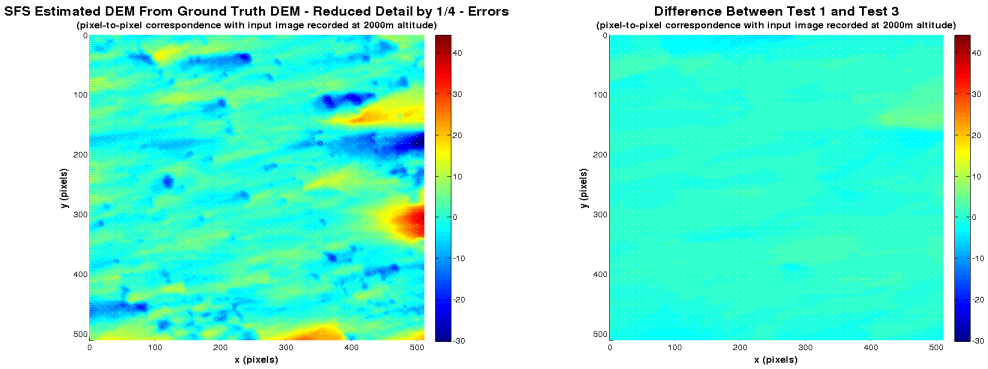


Figure 7.17: Errors in SIRFS results from test using ground truth DEM reduced in quality by $1/4$ (left), and comparison between the SIRFS output from this test and the test on the full ground truth DEM

two tests. This is further validated by the plots presented in Figure 7.17, which shows the errors between the ground truth DEM and the SIRFS output when the input DEM is reduced in quality by $1/4$ (left) and the difference between the SIRFS outputs of test 1 (full GT DEM) and this test (right). In fact, it can be seen from the plot of the difference between this test and the first test (right plot in Figure 7.17), that the output of the SIRFS algorithm on the $1/4$ reduced quality input DEM appears to actually be much better than the difference between the output of the first test and the second test, since the difference plot is almost uniformly very close to zero over the entire surface. There are still a few discrepancies near the image borders, but even these are much less pronounced than in the second test. This may be a consequence of a much more optimal set of input tuning parameters, which were arrived at through trial and error, which by no means guarantees that optimal parameters will be found, but in this case optimality may actually be close to being achieved. However, from examining the errors much more closely, the SIRFS DEM for this test can be seen to have approximately 23% of its surface in error by ± 1m, 42% in error by ± 2m, and 56% in error by ± 3m, which is closer to (actually slightly better than) the results obtained with the full quality ground truth depth map, which may further

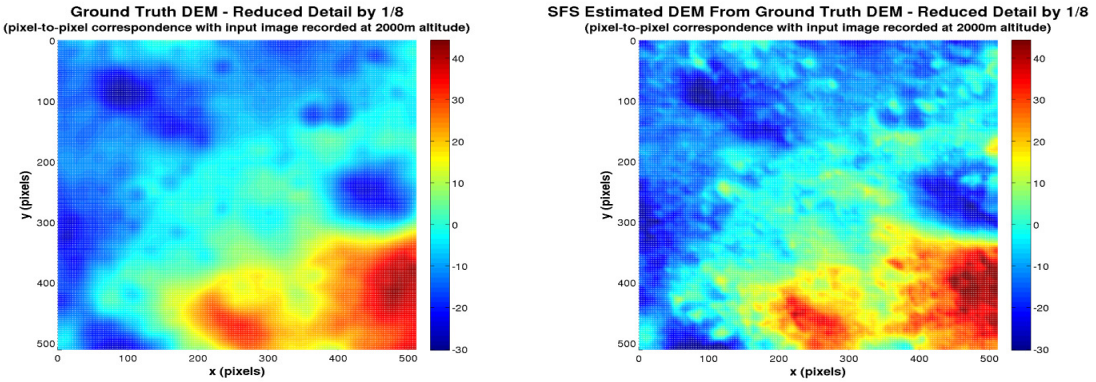


Figure 7.18: SIRFS test using ground truth DEM reduced in detail by $1/8$: Left – input DEM from pyramid level 4 representing a reduction in detail by a factor of $1/8$; Right – SIRFS output DEM

suggest a reduction in over-fitting; and in terms of RMS error this is now slightly worse than that of test 1: 6.06m for test 1 compared to 6.08m for this test, which is a reflection of the extreme errors being slightly greater and/or being slightly larger in terms of region area when compared to the ground truth results.

Figure 7.18 presents the SIRFS results for an input depth map derived from the ground truth DEM with a reduction in quality by a factor of 8. On the left of the figure is the input DEM, which now shows a reasonably clear reduction in quality compared to the original ground truth, and on the right is the DEM produced from the output of the SIRFS algorithm. These results again appear to be highly consistent with the previous results in that they seem to closely match those produced from the test on the full ground truth DEM, However, from the plots of the errors and difference in SIRFS output between this test and the first test on the full ground truth, shown in Figure 7.19, it can be seen that the accuracy is now somewhat reduced from the previous tests. The error plot on the left of the figure still shows a significant proportion of the surface is recovered very accurately, approximately 10%, 20% and 54% of the surface is recovered to within ± 1m, ± 2m and ± 6m, respectively. However, there are a number of regions that have errors beyond ± 20m and even 2 small regions where the errors are around $+40$m. This is reflected in the plot on the right, which shows the difference between the SIRFS results from that using the full ground truth as input and the SIRFS results from this current test. Now we can see that there are regions where the difference is significant, as much as ± 30m in certain places, and these areas are very similar in shape and size to the high error regions in the error plot on the left.

Figure 7.20 shows the input DEM (left) produced from the full ground truth DEM by reducing it in size by a factor of 16 and then up-scaling back to the original size using bilinear interpolation, along with the SIRFS results (right) using this DEM as the initial estimate of the terrain shape. From the results on the right, we can still see that on a large scale the shape is roughly correct even if it is somewhat distorted, but there are now many small areas with significant error. The input DEM, which is now clearly of very low quality compared to the original ground truth DEM, must now be close to the limit of the capability of the SIRFS algorithm, however it may still be possible of improving these

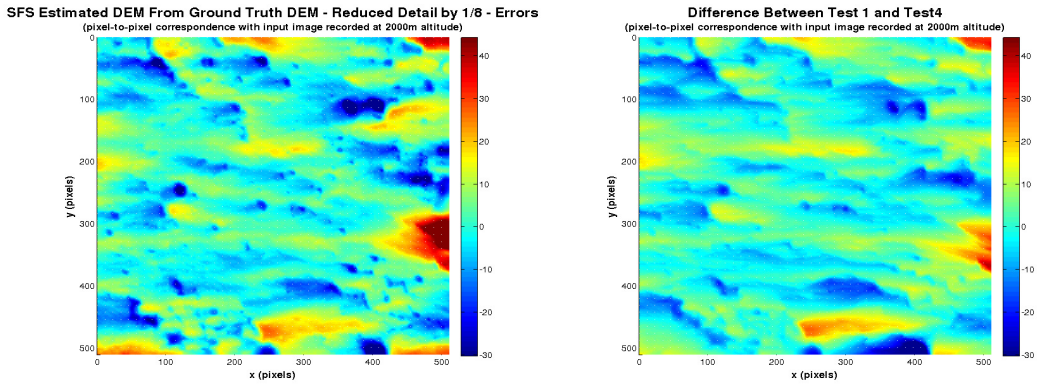


Figure 7.19: Errors in SIRFS results from test using ground truth DEM reduced in quality by $1/8$ (left), and comparison between the SIRFS output from this test and the test on the full ground truth DEM

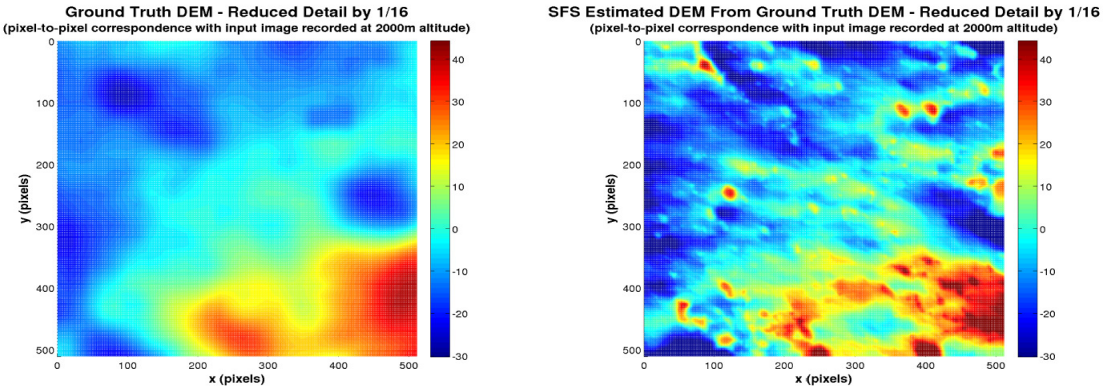


Figure 7.20: SIRFS test using ground truth DEM reduced in detail by $1/16$: Left – input DEM from pyramid level 5 representing a reduction in detail by a factor of $1/16$; Right – SIRFS output DEM

results by additional effort put into finding better values for the tuning parameters as it cannot be guaranteed that those that were used were optimal, despite a fair amount of experimentation with these parameters having been carried out. The results presented here are the best that were obtained from the experimentation carried out on the values of the tuning parameters, but it is still possible that better results could be obtained. The error plots in Figure 7.21 present the actual errors between the SIRFS result and the ground truth (left) and the difference between the SIRFS output that used the full ground truth as the input surface shape and the SIRFS output of this current test (right). From the error plot on the left we can see that there are still a significant number of areas that are estimated to high accuracy: approximately 10%, 19% and 50% of the surface is recovered to within ± 1m, ± 2m and ± 6m, respectively, which is a slight reduction in accuracy from the results of the previous test. The rms error for this test is 11.75m, which is again slightly worse than the previous test, and is a reflection of the regions of high error being visibly larger in both the error plot and the difference plot.

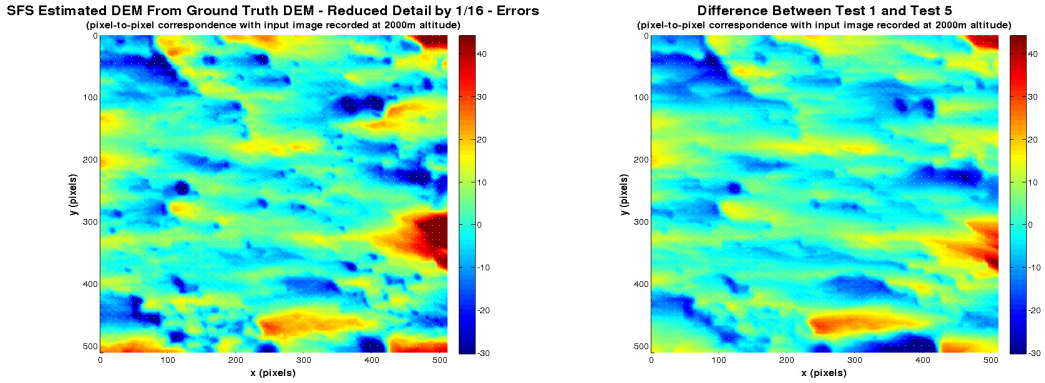


Figure 7.21: Errors in SIRFS results from test using ground truth DEM reduced in quality by $1/16$ (left), and comparison between the SIRFS output from this test and the test on the full ground truth DEM

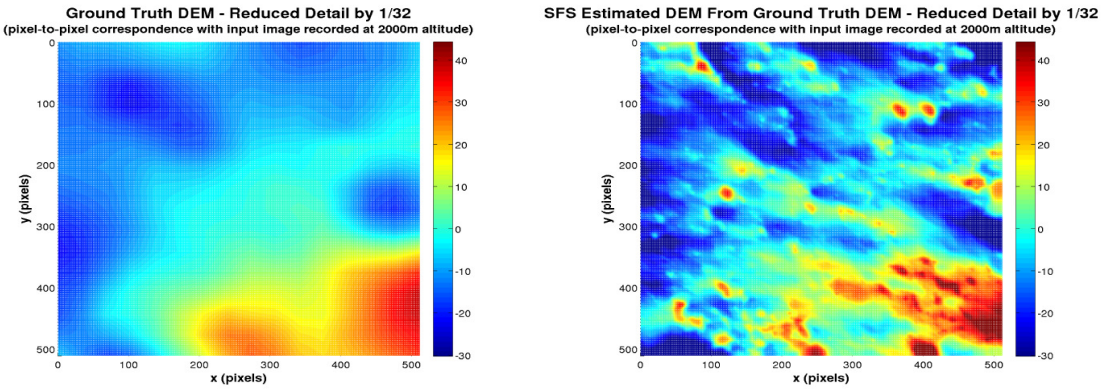


Figure 7.22: SIRFS test using ground truth DEM reduced in detail by $1/32$: Left – input DEM from pyramid level 6 representing a reduction in detail by a factor of $1/32$; Right – SIRFS output DEM

In Figure 7.22, we present the results for a test of the SIRFS algorithm using an input depth-map derived from the ground truth DEM after reducing it in size by a factor of 32 and then up-scaling it back to the original size. The input DEM (left) is now of extremely low resolution, and is even visibly much lower than in the previous test. The SIRFS results on the right, on the whole are still reasonably similar to those of the previous test, however there are some clear differences from the results of the previous test in the region of the lower right quarter of the DEM, particularly in the range of around $(x_w, y_w) = (200, 400)$ to $(400, 500)$. The error plots shown in Figure 7.23 a clear growth in size of the regions of high error compared to those in the previous test, which is reflected in the rms error being 13.76m, however, there are still a number of reasonably large regions of very low error, for example 7% of the DEM is within ± 1m of the ground truth, 14% is in error by ± 2m, and 52% of the surface is within ± 8m of the ground truth.

For completeness, a plot of the rms errors for each of these tests is presented in Figure 7.24, which further illuminates an important result that was briefly mentioned

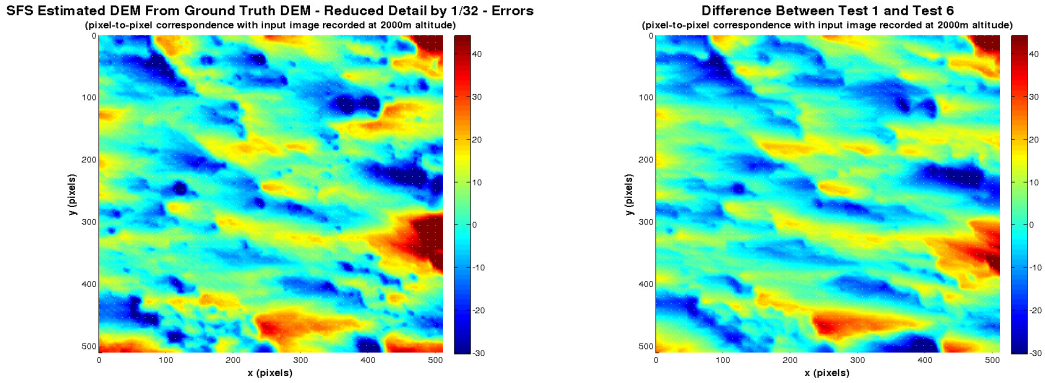


Figure 7.23: Errors in SIRFS results from test using ground truth DEM reduced in quality by $^1/_{32}$ (left), and comparison between the SIRFS output from this test and the test on the full ground truth DEM

above– test 2, which was for the input depth map derived from the full ground truth DEM scaled down in size by a factor of 2 and then up-scaled back to full size using bilinear interpolation, actually resulted in a lower RMS error than the input depth map derived directly from the full ground truth DEM (test 1). Therefore it would appear that some degradation in the quality of the input depth map actually leads to better performance, but this may simply be due to a less optimal set of input tuning parameters for test 1 compared to those used in test 2, so perhaps this is not as significant as it may appear. This may lend further support to the remark made above about the SIRFS results possibly exhibiting over-fitting for test 1, which was assumed to be a result of suboptimal tuning parameters. The difference between the SIRFS output for test 1 and test 3 (shown in Figure 7.17) is also better explained from the plot of RMS errors – it actually has a very similar RMS error to that of test 1. Therefore, the test with the best results is actually test 2, and not test 3, unlike what the previous figures seemed to indicate. In tests 4-6, the rms errors are significantly higher than those of tests 1-3, and show an increase in error with each subsequent test, which is in line with expectation and in agreement with what the previous figures indicated.

Finally, as a preliminary to the work presented in the next chapter, we now move on to assess the performance of the SIRFS algorithm when a sparse input depth map that does not cover the entire field of view of the input image is used as the initial input Z^0 – i.e. we use the interpolated (but not extrapolated) SFM estimated DEM to produce the input depth map (with NaNs inserted in place of any missing pixels) and use this along with the image presented in Figure 7.11. The results from this test are presented in Figure 7.25, from which it can be seen that the SIRFS algorithm performs extremely poorly when there is missing initial input data within the field of view of the image. It is therefore clear the solution to combining the results of SFM and SIRFS is not as straight forward as simply inserting the SFM DEM derived depth map as the initial structure estimate Z^0 in the SIRFS algorithm. Chapter 8 presents the strategy that is used to combine the two techniques to achieve more satisfactory performance.

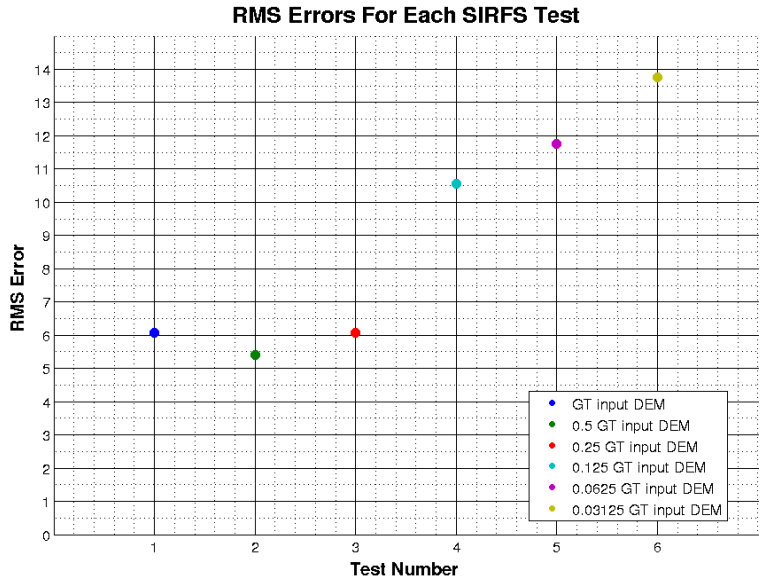


Figure 7.24: RMS errors for each SIRFS test

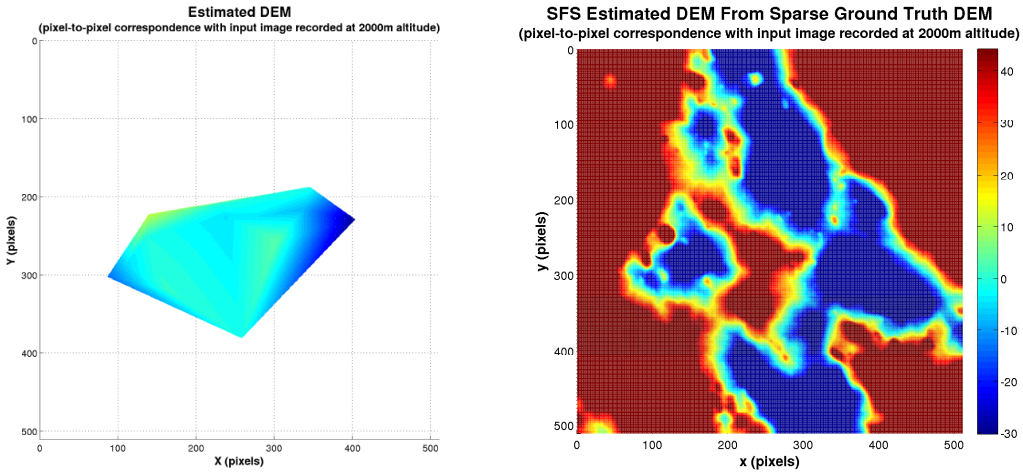


Figure 7.25: SIRFS test using a sparse PSO-MS-SRUKF SFM estimated DEM (left) to derive the input depth map for the SIRFS algorithm, alongside the resulting SIRFS estimated DEM (right)

7.10 Conclusions From SIRFS

A detailed analysis of the performance of the SIRFS algorithm, proposed by Barron and Malik [167, 168], has been carried out using a synthetic PANGU image at an altitude of 2000m and various quality input depth maps derived from the ground truth DEM of the terrain within the field of view of the PANGU input image. From the tests performed on this algorithm it is clear that it shows great potential in producing highly accurate dense DEMs even when the quality of the input DEM is reasonably low. However, it did not

produce satisfactory results when supplied with an input depth map of the sparse type estimated by SFM, which means that a combined approach may require an initial low resolution input DEM that covers the entire field of view of the first descent image. It is possible that such a DEM may be available from orbital surveys from previous or even the current mission, since it is reasonably common practice to record measurements that are suitable for generating such DEMs from an orbiting platform in order to provide the necessary information required to identify scientifically interesting landing sites, and since it is assumed in this work that it is most likely that the descending spacecraft is aiming for a predetermined location, this requirement may not be too restrictive. It is expected that the DEMs produced from a successful combined approach should be of sufficient accuracy to meet the requirements of this project.

8 COMBINING SFS AND SFM

This chapter focusses on the development of a strategy for combining the SIRFS SFS algorithm presented in the preceding chapter with the previously described PSO initialised master-slave square-root unscented Kalman filter based SFM algorithm. From the results obtained in Chapter 7 it is clear that the small and sparse SFM DEMs obtained so far cannot be used straightforwardly to derive input depth maps for the SIRFS algorithm because it seemingly does not handle missing information very well at all. Therefore, in this chapter we assume a low resolution input DEM, obtained from an orbital platform, is available to be used as the input depth map to SIRFS, so that there is no missing data, and from then onwards the goal is to recursively incorporate results from SFM and periodically perform additional SIRFS estimates of the structure of the terrain in order to eventually produce a DEM of sufficient accuracy and density to achieve the goals of this project.

8.1 Strategy for Combining SFS and SFM

Assuming that a low resolution input DEM is available, most likely from an orbital survey that was used to identify potential landing site locations, either from a previous mission or from an earlier stage of the current mission before the landing craft was detached from an orbiting platform to begin its descent. This DEM should be produced so that it completely covers the field of view of the first descent image recorded at an altitude of around 2000m. With this in place we can then run the SIRFS algorithm to refine the initial low resolution DEM and then use the results from this to initialise the SFM algorithm. The SFM results can then be used to further refine the DEM, and after this the SIRFS algorithm can be used again, but this time on a later (closer) image, which should allow for greater surface detail to be discerned. The overlapping region between the initial image (and DEM) and the later image could then be updated and the resulting DEM used again to initialise another stage of SFM estimation. This whole process can be repeated over the entire trajectory so that both methods are used effectively in refining the initial DEM to a level of accuracy and resolution that would be sufficient for achieving the hazard detection requirements of this project, i.e. slopes greater than 5 degrees and surface objects such as rocks that are larger than 30cm.

To further clarify this procedure, we now break it down and present it as a series of steps.

8.1.1 Combination Procedure

1. Produce a depth map at the initial altitude and orientation (i.e. in the world frame) from elevation data supplied in the initial DEM.

2. Run SIRFS algorithm using this depth map and the first image of the descent sequence to produce a refined DEM with greater surface detail.
3. Initialise SFM algorithm using depth information derived from the SIRFS-modified DEM.
4. Update DEM with SFM results.
5. Determine correspondence between the surface region covered by the camera FOV, for the final image used in the SFM algorithm, and the original DEM.
6. Re-run SIRFS algorithm on the final SFM image and corresponding region of DEM.
7. Update corresponding region of DEM with SIRFS output.
8. Repeat from step 4, until the end of the image sequence.

8.2 Results

We now present a set of results for the first 4 steps of the above procedure. This serves as an illustration of what can be achieved by using this combined procedure with only the first short segment of the descent trajectory. Unfortunately due to time constraints and the complexity of determining the correspondence between the initial DEM and the images/DEMs produced from lower altitudes, the remaining steps have not yet been implemented, and so we leave this for further work. The input descent image and chosen initial DEM are presented in Figure 8.1, where the input image (left) is recorded at at altitude of 2000m above the synthetic PANGU surface and the input DEM (right) is that derived from the full ground truth DEM scaled down by a factor of 8, which was chosen because this was the first of the images that were a challenge to the SIRFS algorithm in Chapter 7. Although it was concluded that the SIRFS algorithm performed well in the previous chapter using the degraded ground truth truth DEMs, those tests

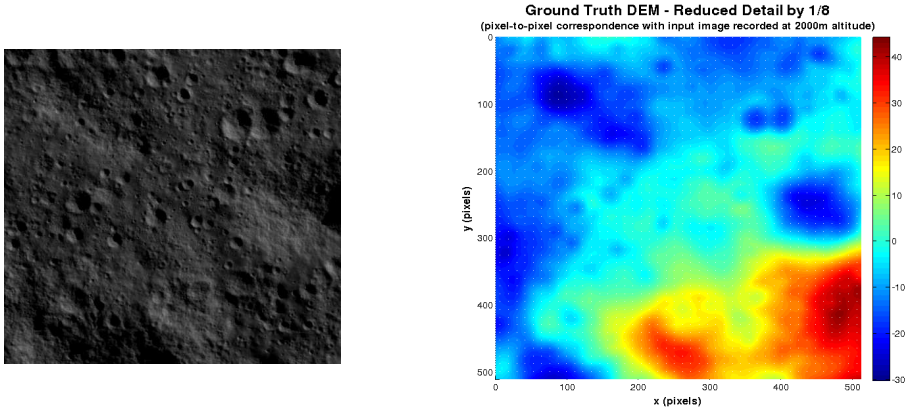


Figure 8.1: Input image (left) and DEM (right) used in first initial test of SFS-SFM combination procedure

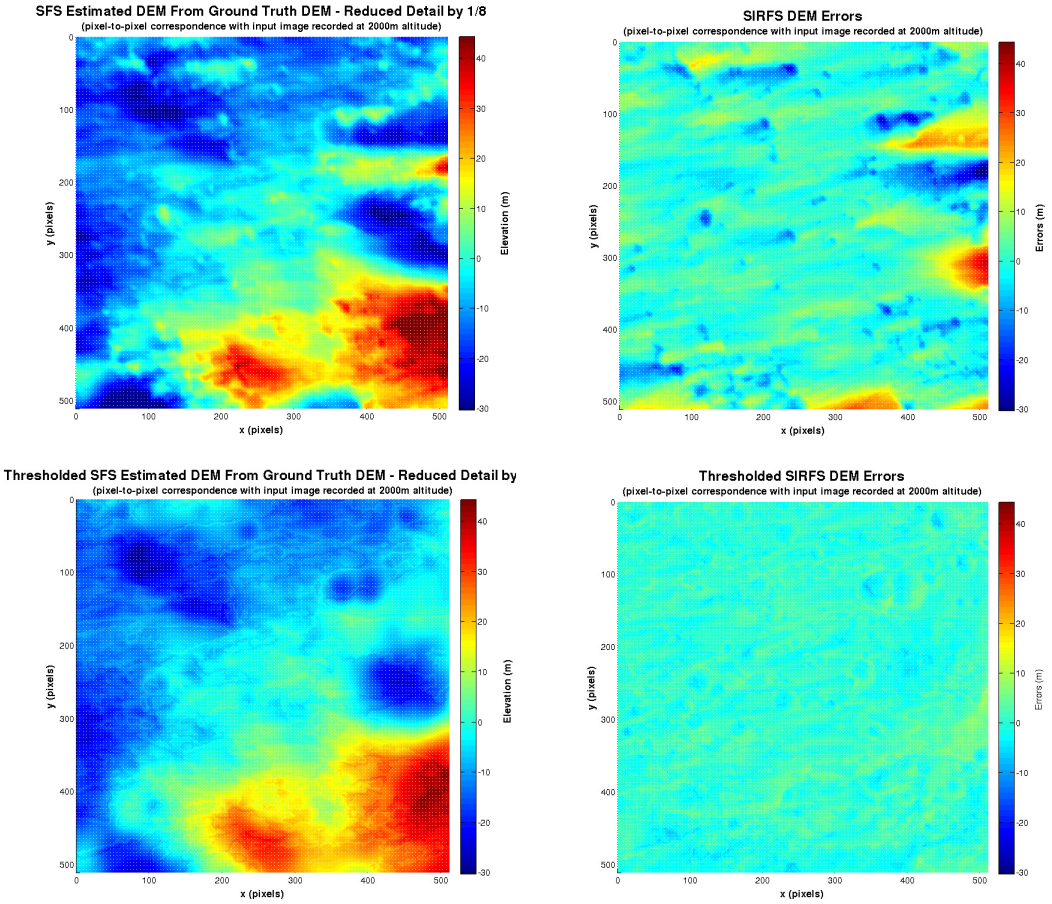


Figure 8.2: Top: Refined DEM obtained directly from SIRFS DEM based on the low-resolution input DEM; Bottom: Refined DEM obtained from thresholding the SIRFS DEM based on the low-resolution input DEM

were carried out only to gain an appreciation of the capabilities of the SIRFS algorithm. At that stage it was not envisaged that in a combined approach we would be assuming the availability of an initial low-resolution DEM that covered the entire area of the initial image. However, since we now are assuming the availability of such a low-resolution DEM, we can use this initial depth information to apply some constraints to the SIRFS algorithm to improve the performance. At this point, this is achieved by applying a simple threshold, in which we only accept the result from the SIRFS algorithm for an individual pixel if the absolute difference between the original input DEM and the SIRFS DEM for that pixel is less than a threshold value (4m was used as the threshold value in the results shown below). By using this thresholding method, the large errors seen in some regions of the SIRFS DEM (which caused visible shape distortions in extremum regions and spurious errors in other small regions) are eliminated and a much better result is achieved overall. Figure 8.2 presents the results from the SIRFS algorithm using the input image and the initial low resolution DEM shown in Figure 8.1, along with the surface errors with respect to the full ground truth DEM, both with (bottom) and without

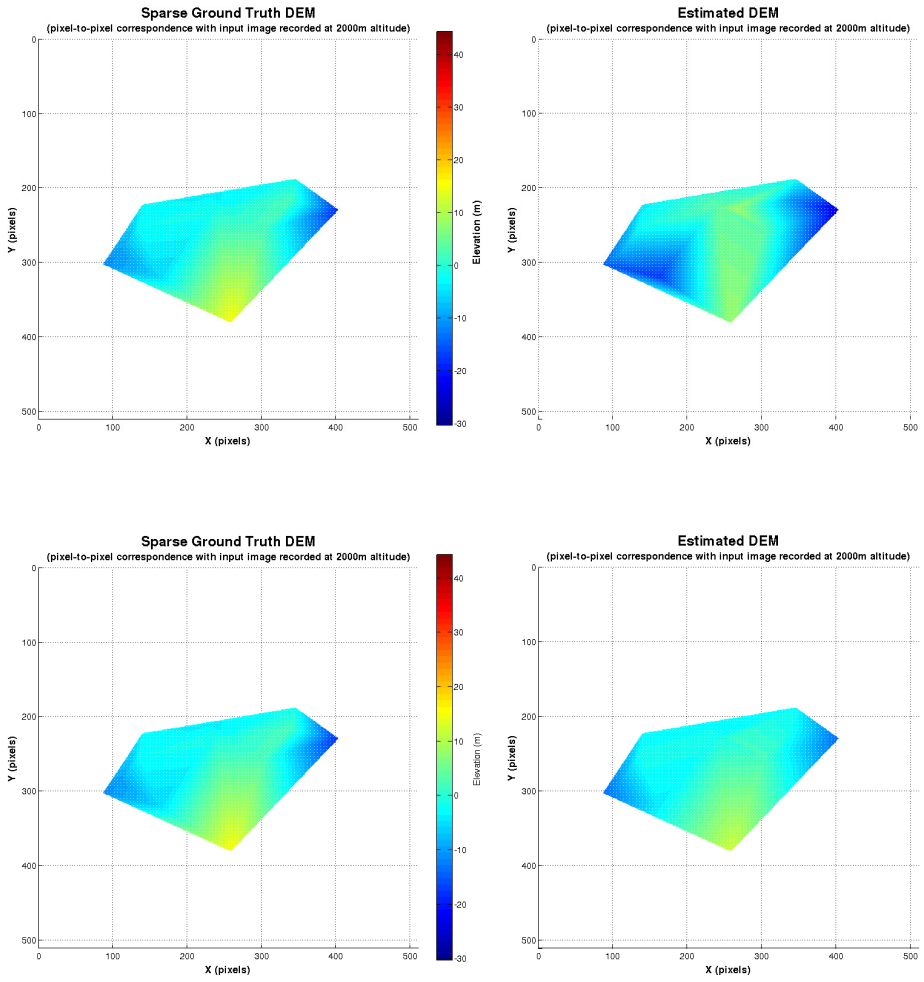


Figure 8.3: Top: Original input DEM initialised SFM Results – Left: Sparse ground truth DEM; Right: SFM estimated DEM. Bottom: SIRFS DEM initialised SFM Results – Left: Sparse ground truth DEM; Right: SFM estimated DEM

(top) this thresholding operation. Here it can very clearly be seen that the thresholded SIRFS results are in much closer agreement with the full resolution ground truth DEM. In the thresholded DEM all the extreme error regions (either red or dark blue) have vanished, and in fact the maximum error is only around 10.6m, whereas in the non-thresholded DEM the maximum error is around 40.7m. Additionally, for the thresholded DEM approximately 41% of the surface has an absolute error below 1m and over 98% of the surface has an absolute error of below 5m, whereas for the non-thresholded DEM it is 21% and 41%, respectively. Finally, we observe an overall RMS error of around 6.49m for the non-thresholded DEM and only 1.98m for the

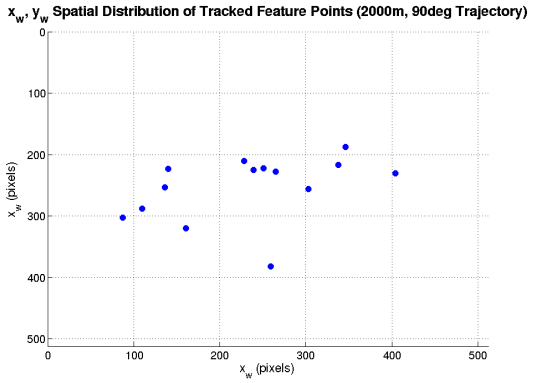


Figure 8.4: Distribution of tracked feature points in the 1st image of SFS-SFM combination tests

thresholded DEM. Therefore the improvements offered by applying this threshold are very significant.

Figure 8.3 presents the results obtained from the PSO-initialised (excluding feature point depths) MS-SRUKF SFM algorithm when both the original low-resolution input DEM was used to initialise the feature point depths (top), and when the thresholded SIRFS DEM was used to initialise the feature point depths (bottom). In both these cases, 14 feature points were tracked over 120 images using the TR-KLT feature tracker. The sparse ground truth DEM is shown in the left column of Figure 8.3 and the SFM estimated results are shown in the right column, and where in all cases the DEM is interpolated in order to fill in the gaps between the tracked feature points, the distribution of which is shown in Figure 8.4 in order to give an indication of the sparsity of the DEMs. These results clearly indicate that the SIRFS initialised SFM results are considerably better than those that were initialised off of the original input DEM, and this is reflected in Figure 8.5, where it can be seen that the errors for the SIRFS initialised SFM DEM are closer to zero over much more of the interpolated surface and has much less variation. This improvement is also seen in the RMS errors, which are 2.1m and 6.3m for the SIRFS initialised SFM and original DEM initialised SFM, respectively. The peak errors are also reduced, with the SIRFS initialised SFM resulting in a maximum error of around 4.7m and the original DEM initialised SFM having a maximum error of 12.4m.

Finally, in Figure 8.6 we present the full DEM resulting from the incorporation of the thresholded SIRFS initialised SFM results into the thresholded SIRFS DEM, alongside the original thresholded SIRFS DEM. While it is not possible to actually see the difference from this figure, nor would it be from a full difference plot between the two DEMs, we can report that the RMS error between the full ground truth DEM and the SFM updated, thresholded SIRFS DEM is 1.97726m and that for the thresholded SIRFS DEM before the incorporation of the SFM results was 1.97735m. Therefore there is a slight improvement. However, while this difference is very small, it is at least an indication that there is merit to combining the two techniques, and if the combination procedure was repeated for the remainder of the trajectory, it is possible that the improvement could become significant to the point where it is clear that a combined solution provides much better performance than, for example, using SIRFS on its own.

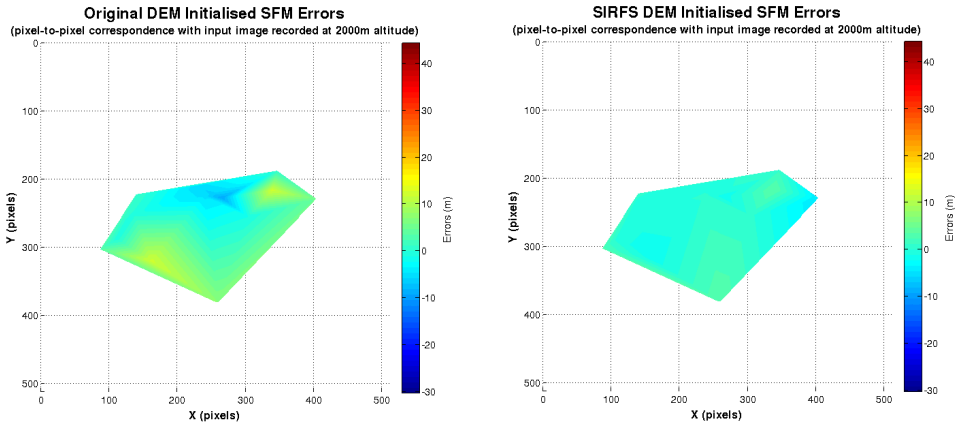


Figure 8.5: DEM Errors for original DEM (left) and SIRFS DEM (right) initialised SFM

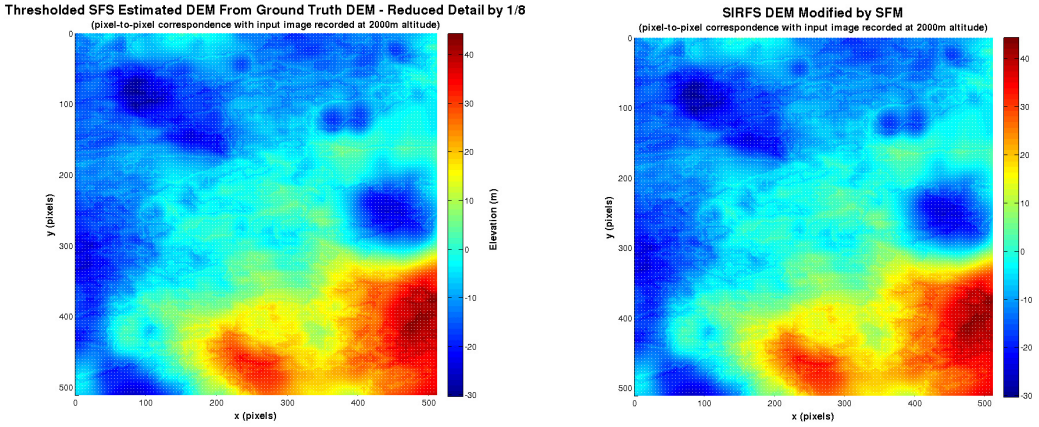


Figure 8.6: Comparison between thresholded SIRFS DEM (left) and thresholded SIRFS DEM with incorporated SFM Results (right)

8.3 Conclusions on Combining SFS and SFM

This chapter has presented a strategy for combining the SIRFS SFS algorithm with the PSO initialised MS-SRUKF SFM algorithm. Unfortunately, direct incorporation of the small sparse SFM DEMs, that were obtained in the preceding chapters without any prior knowledge of the terrain shape, could not be directly incorporated. Therefore an assumption had to be made that an initial low resolution DEM covering the entire field of view of the first descent image is available from some previous orbital observations, which is not an unrealistic requirement for many space missions. Using this initial low resolution DEM, a slight improvement was observed in combining SIRFS with SFM, over the use of SIRFS alone. While the test presented in this chapter only represents part of the overall combination procedure, which could not be fully tested due to complexity and time constraints during the final stages of the project, it provides confidence that multiple iterations of the procedure could enable the accuracy and density of the DEM to be

reasonably significantly improved. Therefore it is expected that the proposed procedure could potentially provide the required level of resolution and accuracy to reliably construct a sophisticated hazard detection system for next generation space landers to detect rocks and slopes that are too steep for a safe landing.

9 CONCLUSIONS AND FURTHER WORK

9.1 Conclusions

This thesis has explored a number of techniques aimed at 3D scene recovery from sequences of monocular images that represent descent imagery recorded during the short period of time under which a spacecraft is descending towards a planetary surface. The focus has been on identifying suitable methods and increasing estimation robustness in order to provide the accuracy that would be required in a hazard detection system that is capable of supporting pin-point landing operations on next generation robotic landing craft. The most significant contribution in this thesis was the development of a self initialising, fully adaptive feature based structure from motion algorithm that was demonstrated to provide highly accurate structure estimates from high, and therefore challenging, altitudes. Although the algorithm was capable of autonomously adapting to a potentially changing noise environment, which may be reasonably expected under the application of planetary descent where varying or perhaps even intermittent rocket thrust or sudden atmospheric turbulence may occur during the descent leading to varying degrees of vibration of the platform and thus varying amounts of image motion blur affecting the accuracy of feature tracking, this capability was not fully tested in this thesis and is therefore left as possible future work. An additional contribution in this thesis was the design of a strategy for combining the SFM structure estimates with that of a very capable shape from shading algorithm, however, due to time constraints at the end of the project this could also not be fully tested.

In Chapter 3 an investigation into feature tracking methods was carried out in order to identify a suitably robust method to use in the SFM algorithm. An investigation into the popular and highly praised SIFT feature tracking method was conducted, which was found to be capable of identifying and reliably tracking an enormous amount of features for long periods of time, which could be extremely beneficial for producing the dense DEMs required by this project. However, it was determined that SIFT could not be employed successfully in this work due to possible feature localisation accuracy issues that result from its foundation as a blob detection method, which unfortunately are not the best types of features to use in 3D reconstruction tasks, especially when strict requirements are placed on the reconstruction accuracy as is the case in this project. Following this, an investigation into the KLT detector was carried out, which is a feature tracking method that tracks more corner-like features that have a very well defined location in the images, and are therefore very suitable for reconstruction tasks. It was determined that the standard KLT algorithm can suffer from significant error growth over long image sequences, which then lead to the development of a more robust version of the KLT algorithm which includes a constraint known as time-reversibility which ensures that the same feature point positions would be found if one was to run the tracker backwards in time as those that are found when the tracker is run forwards in time. This was shown to offer much greater feature tracking stability, allowing the features to be

tracked for far longer than the standard approach, and to have better accuracy. It was therefore decided that this TR-KLT algorithm would be employed in the SFM algorithms.

Chapter 4 carried out a preliminary investigation into SFM methods, and resulted in the development of a robust formulation of a feature-based SFM method that was built upon a filtering framework. The enhanced robustness offered by this approach came mostly from the use of an alternative camera model, which allows for an effective means of decoupling the focal length from the depth and results in only a single structure parameter needing to be estimated for each tracked feature point instead of 3 for other types of methods. It also allows for the focal length to be simultaneously estimated, which means that the algorithm can be applied to (partially) uncalibrated cameras. On top of this, an additional data fusion filter was included that fuses the SFM estimates with measurements from an inertial measurement unit, which then enables the direct estimation of a scale factor that can be used to provide an effective means of overcoming the scale ambiguity problem in monocular SFM, when camera motion is also simultaneously estimated. This method was found to produce highly accurate estimates of both the motion and structure parameters. However, following this preliminary investigation, under the advice of ESA it was assumed that motion would be provided by some other spacecraft subsystem and so there was no need to continue simultaneously estimating the motion parameters, and therefore in the remaining chapters it was assumed that the motion was fully known, which eliminates the need to determine image scale because the scale ambiguity problem disappears when motion is fully known. The filtering framework used in this chapter was based on the well known extended Kalman filter, in order to reduce complexity during the development of the SFM algorithm, but was still able to produce very accurate results.

In an effort to improve robustness and achieve greater accuracy, a number of alternative filtering methods were investigated in Chapter 5, that are reportedly more robust than the EKF. Following on from the results of the previous chapter, a H_∞ filtering framework was investigated that also simultaneously estimated the motion parameters. This was found to offer an improvement in some (but not all) of the motion parameter estimates, but did not improve upon the structure estimates compared to the EKF. Here we also investigated a different method of KLT feature tracking that utilised measurements from an inertial measurement unit to update the initial guess displacement for each feature point prior to the iterative KLT localisation stage of the tracking. While this did allow for greater numbers of features to be tracked for longer than standard KLT, it did nothing to prevent the accumulation of errors, which could now possibly be even worse due to the influence of errors in IMU measurements, and it was ultimately determined to be too inaccurate to be used in its present form for the task of reliable structure estimation. Following this, the development of a PSO initialised, fully adaptive SFM algorithm was undertaken, which was built upon a master-slave square-root unscented Kalman filtering framework, where the master filter is responsible for estimating the structure parameters (now under fully known motion) and the slave filter is responsible for adaptively adjusting the tuning parameters in the event of a change in the noise environment. The slave filter also had additional adaptive mechanisms included (based on covariance matching techniques) so that its tuning parameters could be adaptively adjusted if necessary. The whole algorithm was

initialised using a particle swarm optimisation method in order to try and find an optimal set of initial tuning parameters. The development of this algorithm forms the major contribution of this thesis and is the main source of innovation in this work. It was demonstrated that this algorithm could reliably produce accurate estimates of the structure parameters in the tests presented in this chapter and those included in other chapters. However, due to the complexity of the algorithm, which required a hugely significant development effort, the advanced adaptive capabilities of the algorithm were not fully tested in this work due to time constraints, and so we leave this to future work, but should be noted that these capabilities could prove extremely useful in this application since it is likely that changes of the noise environment will occur, therefore the developed algorithm has the potential to be extremely capable of operating in a wide variety of situations.

Chapter 6 described the acquisition of a number of real image datasets during a 5 month visit to ESA-ESTEC in The Netherlands. These datasets were captured from a test bench set-up consisting of a scaled model of a portion of the real Lunar surface along with a motorised camera platform that enables the camera to be moved very precisely in order to capture highly representative descent images under realistic lighting conditions. These image datasets were used to further test the PSO-MS-SRUKF SFM algorithm, which was found to produce results of a similar accuracy to those obtained from synthetic PANGU image sequences. It was also observed that the feature tracking algorithm performed much better on these real images, which allowed for the features to be reliably tracked for many more images compared to the PANGU images, which would potentially allow for better convergence of the structure parameters. However, difficulty was encountered in mapping the estimated structure parameters to the correct locations within the ground truth DEM, and so this was a source of errors that unfairly degraded the quality of the results. It was therefore decided to continue with the sole use of synthetic PANGU images in the remaining chapters of this thesis.

Chapter 7 investigates a selection of shape from shading techniques, beginning with simple classical methods and later moving on to more sophisticated methods. It was found that the simple classical techniques performed extremely poorly on the complex scenes present in the representative descent imagery, due to the very restrictive assumptions made in these early methods: simple smooth shapes with uniform and fully known albedo are generally assumed in these approaches. Therefore the classical methods were of no use under the application area of this thesis. A very simple slope from shading method was also developed that seemed to show promise and allowed for the direct production of hazard maps. However, in an attempt to achieve greater accuracy, a sophisticated, modern method of shape from shading was investigated that was designed specifically for use on the types of natural images produced from planetary surfaces and so it was directly applicable to descent imagery. It also provides for an elegant means of incorporating prior low resolution depth information such as that produced by SFM algorithms and so this method is very attractive for the application domain of this thesis. This algorithm, known as shape, illumination and reflectance from shading (SIRFS) also simultaneously estimates the light source direction and is quite probably the first technique that truly allows for variable and unknown albedo amongst the SFS algorithms found in the literature. The SIRFS algorithm was extensively tested

using many varying quality input depth maps derived from a ground truth DEM, and was found to produce very accurate results even with a significant reduction in resolution of the input depth maps. However, it was found to be incapable of dealing with missing structure information, therefore it could not be straightforwardly used with the small, sparse DEMs produced from the SFM algorithm. Therefore, Chapter 8 focussed on the development of a strategy for combining the results of SFM and the SIRFS algorithm to produce accurate and dense DEMs of potentially sufficient resolution to meet the requirements of this project. Unfortunately this strategy could not be fully tested due to time constraints at the end of the project, however, it was demonstrated that the combined approach did offer a slight improvement in the resulting DEM for the first estimation step in the descent trajectory.

9.2 Further Work

As has been summarised in the previous section, a large amount of development work has been carried out to devise a suitable approach to tackling the problem of hazard detection during planetary descent. This resulted in the development of an incredibly complex adaptive SFM algorithm, which represents the vast majority of the work undertaken in this thesis. While it has the potential to be extremely capable in handling variations in the noise environment experienced during descent, these advanced capabilities were not fully tested and so their true effectiveness could not be determined. This is an obvious area for future development in other projects. The PSO algorithm used to initialise the MS-SRUKF was also incredibly slow, and although it was able to find a set of tuning parameters that resulted in an extremely low cost, it did not actually achieve full convergence but was instead stopped when one of the particles achieved a satisfactorily low cost, which often required an enormous number of iterations. Therefore the PSO algorithm could benefit from a focussed effort to increase the execution speed, most likely through implementation on dedicated parallel computing hardware, such as a GPU. This could potentially lead to a very significant speed increase since the PSO algorithm is highly parallelisable. With a suitable speed-up, the PSO algorithm may be allowed to run to full convergence in a reasonable amount of time, which may lead to improved accuracy in the results. Alternatively, a different method of initialising the MS-SRUKF could be developed, such as an approach based on neural networks, for example. In its current form, the MS-SRUKF with PSO initialisation could not be used in a real time system, because the PSO can actually take days to complete, which is clearly unacceptable for a planetary descent that may only take 10–15 minutes. However, the PSO part of the overall algorithm would not necessarily be required in a final working system – it could simply be used as a means of finding an optimal set of initial tuning parameters, on the ground prior to mission launch, from representative imagery (instead of manually tuning the filter, which is extremely difficult when dealing with the weakly observable structure parameters), and then the adaptive mechanisms can make any required adjustments to better suit the situation encountered. These adaptive mechanisms along with the actual structure estimation do execute sufficiently quickly to potentially be used in real-time. However, further work would obviously be

required to actually implement a real-time capable system.

Due to time constraints and the complexity of determining correspondence between DEMs and images at different scales, the combined SFM and SFS could only be tested for one run of SFS and SFM (i.e. steps 1-4 of the procedure). Although an improvement from the combination of the two techniques was observed this was only very slight. It is expected that the improvements from applying the procedure over the entire trajectory would accumulate to something significant but much more work is required to firmly establish how the two techniques will perform together over the remainder of the descent trajectory. The influence of the SFM in this procedure is also potentially hindered due to the low numbers of feature points that could be tracked for long enough to produce accurate structure estimates. Even though an improved feature tracker (with respect to the standard KLT tracker) was investigated and improvements were demonstrated, the numbers of features were still very low compared to what other feature trackers can achieve (e.g. SIFT – but this could not be successfully used). It is therefore clear that further investigation is required into increasing the density of the SFM DEMs by experimenting with and perhaps improving upon other available feature trackers. As mentioned, a means of tracking large quantities of features was investigated in the form of the SIFT algorithm, but it was ultimately determined that these could not be reliably used in the SFM algorithm. A detailed investigation into why this seems to be the case is needed, and then perhaps a means of modifying the SIFT algorithm could be devised so that it only tracks the specific types of features that are best suited for 3D reconstruction. Whether this would actually result in more features remains to be seen, but it is possible due to the sheer number of features tracked in its current form. Alternatively, the SFM algorithm could be modified to behave more like a SLAM algorithm by allowing new features to be added at any point during the image sequence, which would also re-find those features that are lost along the way as well as introducing many additional features and consequently producing a much higher density DEM. How this would affect the stability of the filter would be a major focus of this additional work since there would now likely be a significant number of features that are only tracked for a few frames, which would not be sufficient for convergence.

REFERENCES

[1] Thomas Ormston. Time delay between Mars and Earth, May 2012.

[2] R.D. Braun and R.M. Manning. Mars exploration entry, descent, and landing challenges. *Journal of Spacecraft and Rockets*, 44(2):310–323, 2007.

[3] Artem Ivankov. NASA - NSSDC - Spacecraft - Details - Luna 9.

[4] H. Winkelman, J. Gebbie, M. Symonds, and G. Ortega. Entry, Descent and Landing Systems: "The Next Generation". In *European Space Agency, (Special Publication) ESA SP*, Proceedings of DASIA 2003, pages 129–138, Prague, 2003.

[5] M.J. Grant, B.A. Steinfeldt, R.D. Braun, and G.H. Barton. Smart divert: A new mars robotic entry, descent, and landing architecture. *Journal of Spacecraft and Rockets*, 47(3):385–393, 2010.

[6] X. Huang and D. Wang. Autonomous navigation and guidance for pinpoint lunar soft landing. In *2007 IEEE International Conference on Robotics and Biomimetics, ROBIO*, 2007 IEEE International Conference on Robotics and Biomimetics, ROBIO, pages 1148–1153, Yalong Bay, Sanya, 2008.

[7] K. Janschek, V. Tchernykh, and M. Beck. Performance analysis for visual planetary landing navigation using optical flow and DEM matching. In *Collection of Technical Papers - AIAA Guidance, Navigation, and Control Conference 2006*, volume 7 of *AIAA Guidance, Navigation, and Control Conference 2006*, pages 4818–4834, Keystone, CO, 2006.

[8] W. Mahmood and S.M.A. Shah. Vision based hazard detection and obstacle avoidance for planetary landing. In *Proceedings of 2009 2nd International Workshop on Nonlinear Dynamics and Synchronization, INDS 2009*, 2009 2nd International Workshop on Nonlinear Dynamics and Synchronization, INDS 2009, pages 175–181, Klagenfurt, 2009.

[9] T. Brady and J. Schwartz. ALHAT system architecture and operational concept. In *IEEE Aerospace Conference Proceedings*, 2007 IEEE Aerospace Conference, Big Sky, MT, 2007.

[10] C.D. Epp and T.B. Smith. Autonomous Precision Landing and Hazard Detection and Avoidance Technology (ALHAT). In *IEEE Aerospace Conference Proceedings*, 2007 IEEE Aerospace Conference, Big Sky, MT, 2007.

[11] V.M.G. Silva and S.M. Parkes. On the Design of an Optimal GNC Sensor Architecture for Autonomous Planetary Landers. In *European Space Agency, (Special Publication) ESA SP*, Proceedings of DASIA 2003, pages 139–150, Prague, 2003.

[12] A.E. Johnson, Y. Cheng, and L.H. Matthies. Machine vision for autonomous small body navigation. In *IEEE Aerospace Conference Proceedings*, volume 7 of *2000 IEEE Aerospace Conference*, pages 661–671, Big Sky, MT, 2000.

[13] A.E. Johnson, A. Ansar, L.H. Matthies, N. Trawny, A.I. Mourikis, and S.I. Roumeliotis. A general approach to terrain relative navigation for planetary landing. In *Collection of Technical Papers - 2007 AIAA InfoTech at Aerospace Conference*, volume 2 of *2007 AIAA InfoTech at Aerospace Conference*, pages 1498–1506, Rohnert Park, CA, 2007.

[14] J. Li and H. Cui. Vision-aided inertial navigation for pin-point landing on Mars. In *Proceedings of the 2nd International Conference on Intelligent Control and Information Processing, ICICIP 2011*, 2nd International Conference on Intelligent Control and Information Processing, ICICIP 2011, pages 14–18, Harbin, 2011.

[15] B. Van Pham, S. Lacroix, M. Devy, M. Drieux, and C. Phillipe. Visual landmark constellation matching for spacecraft pinpoint landing. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, AIAA Guidance, Navigation, and Control Conference and Exhibit, Chicago, IL, 2009.

[16] B. Van Pham, S. Lacroix, M. Devy, T. Voirin, M. Drieux, and C. Bourdarias. FUSION OF ABSOLUTE VISION-BASED LOCALIZATION AND VISUAL ODOMETRY FOR SPACECRAFT PINPOINT LANDING. In *International Planetary Probe Workshop (IPPW)*, 2010.

[17] C. Yang and A. Ansar. Landmark based position estimation for pinpoint landing on Mars. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2005 of *2005 IEEE International Conference on Robotics and Automation*, pages 4470–4475, Barcelona, 2005.

[18] Eleanor Crane and Steve Rock. Investigations into Vision-Based Hazard Estimation During Autonomous Lunar Landing.

[19] A. Huertas, Y. Cheng, and L. H. Matthies. Automatic hazard detection for landers. In *9th International Symposium on Artificial Intelligence, R & A in Space, Los Angeles, California, February 26, 2008.* Pasadena, CA: Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2008., 2008.

[20] A. Huertas, Y. Cheng, and R. Madison. Passive imaging based multi-cue hazard detection for spacecraft safe landing. In *IEEE Aerospace Conference Proceedings*, volume 2006 of *2006 IEEE Aerospace Conference*, Big Sky, MT, 2006.

[21] P.G. Antreasian, S.R. Chesley, J.K. Miller, J.J. Bordi, and B.G. Williams. The design and navigation of the near Shoemaker landing on Eros. In *Advances in the Astronautical Sciences*, volume 109 II of *Proceedings of the AAS/AIAA Astodynamics Conference*, pages 989–1015, Quebec City, Que., 2002.

[22] W.M. Owen Jr., T.C. Wang, A. Harch, M. Bell, and C. Peterson. Near optical navigation at Eros. In *Advances in the Astronautical Sciences*, volume 109 II

of *Proceedings of the AAS/AIAA Astodynamics Conference*, pages 1075–1087, Quebec City, Que., 2002.

[23] B.G. Williams. Technical challenges and results for navigation of NEAR Shoemaker. *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)*, 23(1):34–45, 2002.

[24] John Cuseo, Paul Fleming, William Hoff, Cheryl Sklair, Monndy Eshera, and Gary Whitten. Machine vision techniques for planetary terminal descent hazard avoidance and landmark tracking. In *Proceedings of the American Control Conference*, volume 2 of *Proceedings of the 1991 American Control Conference*, pages 1406–1407, Boston, MA, USA, 1991. Publ by American Automatic Control Council.

[25] Wolfgang Poelzleitner and Gerhard Paar. Surface-relative spacecraft navigation using computer vision. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 2352 of *Mobile Robots IX*, pages 91–103, Boston, MA, USA, 1995. Society of Photo-Optical Instrumentation Engineers.

[26] Gerhard Paar and Wolfgang Poelzleitner. Autonomous spacecraft navigation using computer vision: a case study for the moon. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 2591 of *Mobile Robots X*, pages 157–169, Philadelphia, PA, USA, 1995. Society of Photo-Optical Instrumentation Engineers.

[27] E. A. Johnson and H. L. Mathies. Precise Image-Based Motion Estimation for Autonomous Small Body Exploration. In *Proceedings of the Fifth International Symposium, ISAIRAS '99*, volume 440, page 627, August 1999.

[28] A. Benedetti and P. Perona. Real-time 2-D feature detection on a reconfigurable computer. pages 586–593.

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

[30] S.I. Roumeliotis, A.E. Johnson, and J.F. Montgomery. Augmenting inertial navigation with image-based motion estimation. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 4 of *2002 IEEE International Conference on Robotics and Automation*, pages 4326–4333, Washington, DC, 2002.

[31] Larry Henry Matthies. *Dynamic stereo vision*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1989. AAI9023429.

[32] M. Bajracharya. Single image based hazard detection for a planetary lander. In *Automation Congress, 2002 Proceedings of the 5th Biannual World*, volume 14, pages 585–590, 2002.

[33] Zhang Zexu, Wang Weidong, and Cui Pingyuan. A reliable algorithm of rock detection and avoidance for safe spacecraft landing. In *Systems and Control in Aeronautics and Astronautics (ISSCAA), 2010 3rd International Symposium on*, pages 1009–1013, June 2010.

[34] W. Hoff and C. Sklair. Planetary terminal descent hazard avoidance using optical flow. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 238–243 vol.1, May 1990.

[35] B. Van Pham, S. Lacroix, M. Devy, M. Drieux, and T. Voirin. Landmark Constellation matching for planetary lander absolute localization. In *VISAPP 2010 - Proceedings of the International Conference on Computer Vision Theory and Applications*, volume 1 of *5th International Conference on Computer Vision Theory and Applications, VISAPP 2010*, pages 267–274, Angers, 2010.

[36] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. E. Johnson, A. Ansar, and L. Matthies. Vision-Aided Inertial Navigation for Spacecraft Entry, Descent, and Landing. *IEEE Transactions on Robotics*, 25(2):264–280, April 2009.

[37] Nikolas Trawny, Anastasios I. Mourikis, Stergios I. Roumeliotis, Andrew E. Johnson, and James F. Montgomery. Vision-aided inertial navigation for pin-point landing using observations of mapped landmarks. *Journal of Field Robotics*, 24(5):357–378, 2007.

[38] Victor Silva and Steve Parkes. Integration of Vision and Navigation Information for Autonomous Planetary Landers.

[39] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[40] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[41] B. P. Amavasai, F. Caparrelli, A. Selvan, M. Boissenin, J. R. Travis, and S. Meikle. Machine vision methods for autonomous microrobotic systems. *Kybernetes*, 34(9/10):1421–1439, 2005.

[42] Q. Zhu, B. Wu, and N. Wan. A sub-pixel location method for interest points by means of the Harris interest strength. *Photogrammetric Record*, 22(120):321–335, 2007.

[43] D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999.

[44] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[45] K. Peng, X. Chen, D. Zhou, and Y. Liu. 3d reconstruction based on SIFT and Harris feature points. In *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, pages 960–964, December 2009.

[46] Bruce D. Lucas and Takeo Kanade. Iterative Image Registration Technique With An Application To Stereo Vision. volume 2 of *Proceedings of the 7th International Joint Conference on Artificial Intelligence.*, pages 674–679, Vancouver, BC, Can, 1981.

[47] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. Technical report, International Journal of Computer Vision, 1991.

[48] Jean-yves Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.

[49] Jean-Yves Bouguet. Pyramidal Implementation of the Affine Lucas Kanade Feature Tracker Description of the Algorithm. *Intel Corporation, Microprocessor Research Labs*, 2001.

[50] Myung Hwangbo, Jun-Sik Kim, and T. Kanade. Inertial-aided KLT feature tracking for a moving camera. pages 1909–1916, 2009.

[51] Jun-Sik Kim, Myung Hwangbo, and T. Kanade. Realtime affine-photometric KLT feature tracker on GPU in CUDA framework. pages 886–893, September 2009.

[52] H. Wu, R. Chellappa, A. C. Sankaranarayanan, and S. K. Zhou. Robust Visual Tracking Using the Time-Reversibility Constraint. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, October 2007.

[53] Rama Chellappa, Gang Qian, and Sridhar Srinivasan. Structure from motion: Sparse versus dense correspondence methods. In *IEEE International Conference on Image Processing*, volume 2, pages 492–499, 1999.

[54] J.K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. *Proceedings of the IEEE*, 76(8):917–935, 1988.

[55] A. Calway. Recursive estimation of 3d motion and surface structure from local affine flow parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):562–574, 2005.

[56] Joachim Heel. Direct Estimation of Structure and Motion from Multiple Frames. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1990.

[57] Yalin Xiong and Steven A. Shafer. Dense structure from a dense optical flow sequence. In *Proceedings of the IEEE International Conference on Computer Vision*, International Symposium on Computer Vision, ISCV'95, Proceedings, pages 1–6, Coral Gables, FL, USA, 1995. IEEE.

[58] Marco Zucchelli. Optical Flow Based Structure from Motion. Technical report, in Numerical Analysis and Computer Science. Stockholm: Kungl Tekniska Hogskolan (Royal Institute of Techology, 2002.

[59] Dah-Jye Lee, Paul C. Merrell, Brent E. Nelson, and Zhaoyi Wei. Multi-frame structure from motion using optical flow probability distributions. *NEUROCOMPUTING*, 72(4-6, SI):1032–1041, January 2009.

[60] TS HUANG and AN NETRAVALI. MOTION AND STRUCTURE FROM FEATURE CORRESPONDENCES - A REVIEW. *PROCEEDINGS OF THE IEEE*, 82(2):252–268, February 1994.

[61] T. Jebara, A. Azarbayejani, and A. Pentland. 3d structure from 2d motion. *Signal Processing Magazine, IEEE*, 16(3):66 –84, May 1999.

[62] Shimon Ullman. *The Interpretation of Visual Motion*. MIT Press, Cambridge, MA, USA, 1979.

[63] JA WEBB and JK AGGARWAL. VISUALLY INTERPRETING THE MOTION OF OBJECTS IN SPACE. *COMPUTER*, 14(8):40–46, 1981.

[64] JW ROACH and JK AGGARWAL. DETERMINING THE MOVEMENT OF OBJECTS FROM A SEQUENCE OF IMAGES. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INLIGENCE*, 2(6):554–562, 1980.

[65] HC LONGUETHIGGINS. A COMPUTER ALGORITHM FOR RECONSTRUCTING A SCENE FROM 2 PROJECTIONS. *NATURE*, 293(5828):133–135, 1981.

[66] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. In *Proceedings of PRIP 82. IEEE Computer Society Conference on Pattern Recognition and Image Processing*, pages 112–18, New York, NY, USA, 1982. IEEE. Proceedings of PRIP 82. IEEE Computer Society Conference on Pattern Recognition and Image Processing, 14-17 June 1982, Las Vegas, NV, USA.

[67] BKP HORN. RELATIVE ORIENTATION. *INTERNATIONAL JOURNAL OF COMPUTER VISION*, 4(1):59–78, January 1990.

[68] J. Oliensis and J.Inigo Thomas. Incorporating motion error in multi-frame structure from motion. In *Proceedings of the IEEE Workshop on Visual Motion*, Proceedings of the IEEE Workshop on Visual Motion, pages 8–13, Princeton, NJ, USA, 1991. Publ by IEEE.

[69] S. Soatto, P. Perona, R. Frezza, and G. Picci. Recursive motion and structure estimation with complete error characterization. In *IEEE Computer Vision and Pattern Recognition*, Proceedings of the 1993 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 428–433, New York, NY, USA, 1993. Publ by IEEE.

[70] Ted J. Broida and Rama Chellappa. ESTIMATION OF OBJECT MOTION PARAMETERS FROM NOISY IMAGES. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):90–99, 1986.

[71] Yt J. Broida, S. Chandrashekhar, R. Chellappa, Zs Ch, T. J. Broida, and S. Ch. Recursive 3-D Motion Estimation from a Monocular Image Sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26:639–656, 1990.

[72] Ali Azarbayejani and Alex P. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):562–575, 1995.

[73] D.Q. Huynh and A. Heyden. Recursive structure and motion estimation from noisy uncalibrated video sequences. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1 –5, December 2008.

[74] Javier Civera, Oscar G. Grasa, Andrew J. Davison, and J. M. M. Montiel. 1-Point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual odometry. *Journal of Field Robotics*, 27(5):609–631, 2010.

[75] B Clipp, G Welch, J M Frahm, and M Pollefeys. Structure from Motion via a Two-Stage Pipeline of Extended Kalman Filters. In *Proceedings of the British Machine Vision Conference*, University of Warwick, September 2007.

[76] D. Nister. Preemptive RANSAC for live structure and motion estimation. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 199 –206 vol.1, October 2003.

[77] Gang Qian, R. Chellappa, and Qinfen Zheng. Robust structure from motion estimation using inertial data. *Journal of the Optical Society of America A (Optics, Image Science and Vision)*, 18(12):2982–97, December 2001.

[78] D. Aufderheide and W. Krybus. A visual-inertial approach for camera egomotion estimation and simultaneous recovery of scene structure. In *2010 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS 2010)*, page 6 pp., Piscataway, NJ, USA, 2010. IEEE. 2010 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS 2010), 6-8 Sept. 2010, Taranto, Italy.

[79] S.-H. Jung and C.J. Taylor. Camera trajectory estimation using inertial sensor measurements and structure from motion results. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, pages II–732–7 vol.2, Los Alamitos, CA, USA, 2001. IEEE Comput. Soc. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 8-14 Dec. 2001, Kauai, HI, USA.

[80] G. Ntzi, S. Weiss, D. Scaramuzza, and R. Siegwart. Fusion of IMU and vision for absolute scale estimation in monocular SLAM. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 61(1-4):287–299, 2011.

[81] Chris J Venter and Ben Herbst. Structure from Motion estimation using a Non-Linear Kalman Filter. In *Proceedings of the 16th Annual Symposium of the Pattern Recognition Association of South Africa*, Langebaan, November 2005.

[82] Mohd Kharbat and Nabil Aouf. Recursive estimation of three-dimensional unmanned aerial vehicle motion and structure based on the L-infinity norm. In *Proceedings of the Institution of Mechanical Engineers*. Journal of Aerospace Engineering, November 2011.

[83] S. Soatto and P. Perona. Reducing structure from motion: a general framework for dynamic vision part 2: implementation and experimental assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(9):943–960, 1998.

[84] A. Fakih and J. Zelek. Structure from motion: combining features correspondences and optical flow. In *ICPR 2008 19th International Conference on Pattern Recognition*, page 4 pp., Piscataway, NJ, USA, 2008. IEEE. ICPR 2008 19th International Conference on Pattern Recognition, 8-11 Dec. 2008, Tampa, FL, USA.

[85] D.W. Way, R.W. Powell, A. Chen, A.D. Steltzner, A.M.S. Martin, P.D. Burkhart, and G.F. Mendeck. Mars science laboratory: Entry, descent, and landing system performance. In *IEEE Aerospace Conference Proceedings*, 2007 IEEE Aerospace Conference, Big Sky, MT, 2007.

[86] Simon J. Julier and Jeffrey K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. pages 182–193, 1997.

[87] G.A. Einicke and L.B. White. The extended H filter-a robust EKF. volume iv, pages IV/181–IV/184 vol.4, 1994.

[88] G.A. Einicke and L.B. White. Robust extended Kalman filtering. *Signal Processing, IEEE Transactions on*, 47(9):2596–2599, September 1999.

[89] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004.

[90] S. Julier, J. Uhlmann, and H.F. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *Automatic Control, IEEE Transactions on*, 45(3):477–482, March 2000.

[91] R. Van der Merwe and E.A. Wan. The square-root unscented Kalman filter for state and parameter-estimation. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on*, volume 6, pages 3461–3464 vol.6, 2001.

[92] Dan Simon. The H-Infinity Filter. In *Optimal State Estimation*, pages 333–372. John Wiley & Sons, Inc., 2006.

[93] U. Theodor, U. Shaked, and C.E. de Souza. A game theory approach to robust discrete-time H-estimation. *Signal Processing, IEEE Transactions on*, 42(6):1486–1495, June 1994.

[94] Luke Feetham, Nabil Aouf, Clement Bourdarias, and Thomas Voirin. Single Camera Absolute Structure from Motion using H-infinity Data Fusion for a Next Generation Planetary Lander. Munich, Germany, July 2013.

[95] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Fluids Engineering*, 82(1):35–45, March 1960.

[96] Paul D. Abramson. *Simultaneous estimation of the state and noise statistics in linear dynamical systems*. PhD, Massachusetts Institute of Technology, May 1968.

[97] I. WEISS. A survey of discrete Kalman-Bucy filtering with unknown noise covariances. In *Guidance, Control and Flight Mechanics Conference*, Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, August 1970.

[98] K. Myers and B.D. Tapley. Adaptive sequential estimation with unknown noise statistics. *Automatic Control, IEEE Transactions on*, 21(4):520–523, August 1976.

[99] Chis Goodall and Naser El-Sheimy. Intelligent Tuning of a Kalman Filter Using Low-cost Mems Inertial Sensors. In *International Society for Photogrammetry and Remote Sensing*, volume XXXVI-5/C55, Padua, Italy, May 2007.

[100] R.K. Mehra. Approaches to adaptive filtering. *Automatic Control, IEEE Transactions on*, 17(5):693–698, October 1972.

[101] David Magill, Thomas. Optimal Adaptive Estimation of Sampled Stochastic Processes, December 1963.

[102] D. Magill. Optimal adaptive estimation of sampled stochastic processes. *Automatic Control, IEEE Transactions on*, 10(4):434–439, October 1965.

[103] R.K. Mehra. On the identification of variances and adaptive Kalman filtering. *Automatic Control, IEEE Transactions on*, 15(2):175–184, April 1970.

[104] R.K. Mehra. On-line identification of linear dynamic systems with applications to Kalman filtering. *Automatic Control, IEEE Transactions on*, 16(1):12–21, February 1971.

[105] Brian J. Odelson, Murali R. Rajamani, and James B. Rawlings. A new autocovariance least-squares method for estimating noise covariances. *Automatica*, 42(2):303 – 308, 2006.

[106] A. H. Jazwinski. Adaptive filtering. *Automatica*, 5(4):475 – 485, 1969.

[107] R.L. Kirlin and A. Moghaddamjoo. Robust adapative Kalman filtering for systems with unknown step inputs and non-Gaussian measurement errors. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 34(2):252–263, April 1986.

[108] F.D. Groutage, R.G. Jacquot, and D.E. Smith. State variable estimation using adaptive Kalman filter with robust smoothing. In *Decision and Control, 1983. The 22nd IEEE Conference on*, pages 1316–1318, December 1983.

[109] F. Mosteller and J.W. Tukey. *Data Analysis and Regression: A Second Course in Statistics*. Addison-Wesley series in behavioral science. Addison-Wesley Publishing Company, 1977.

[110] A. Moghaddamjoo and R.L. Kirlin. Robust adaptive Kalman filtering with unknown inputs. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(8):1166–1175, August 1989.

[111] A. H. Mohamed and K. P. Schwarz. Adaptive Kalman Filtering for INS/GPS. *Journal of Geodesy*, 73(4):193–203, 1999.

[112] Jinling Wang, Mike Stewart, and Maria Tsakiri. Online Stochastic Modelling for INS/GPS Integration. In *12th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 1999)*, pages 1887–1896, Nashville, TN, September 1999.

[113] Ahmed El-Mowafy and Ahmed Mohamed. Attitude Determination from GNSS Using Adaptive Kalman Filtering. *Journal of Navigation*, 58(01):135–148, 2005.

[114] Christopher HIde, Terry Moore, and Martin Smith. Adaptive Kalman Filtering for Low-cost INS/GPS. *The Journal of Navigation*, 56(01):143–152, 2003.

[115] C. Hide, T. Moore, and M. Smith. Adaptive Kalman filtering algorithms for integrating GPS and low cost INS. In *Position Location and Navigation Symposium, 2004. PLANS 2004*, pages 227–233, April 2004.

[116] W Ding, J Wang, and C Rizos. Stochastic modelling strategies in GPS/INS data fusion process. In *International Global Navigation Satellite Systems Society*, Surfers Paradise, Australia, July 2006.

[117] W Ding, J Wang, and C Rizos. Improving covariance based adaptive estimation for GPS/INS integration. pages 259–294, Jeju, Korea, October 2006.

[118] Weidong Ding, Jinling Wang, Chris Rizos, and Doug Kinlyside. Improving Adaptive Kalman Estimation in GPS/INS Integration. *Journal of Navigation*, 60(03):517–529, 2007.

[119] Ali Almagbile, Jinling Wang, and Weidong Ding. Evaluating the performances of adaptive Kalman filter methods in GPS/INS integration. *Journal of Global Positioning Systems*, 9(1):33–40, 2010.

[120] A.A. Girgis and W.L. Peterson. Adaptive estimation of power system frequency deviation and its rate of change for calculating sudden power system overloads. *Power Delivery, IEEE Transactions on*, 5(2):585–594, April 1990.

[121] L. Jetto, S. Longhi, and G. Venturini. Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots. *Robotics and Automation, IEEE Transactions on*, 15(2):219–229, April 1999.

[122] M. Ficocelli and F. Janabi-Sharifi. Adaptive filtering for pose estimation in visual servoing. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 19–24 vol.1, 2001.

[123] Vincenzo Lippiello, Bruno Siciliano, and Luigi Villani. Adaptive extended Kalman filtering for visual motion estimation of 3d objects. *Control Engineering Practice*, 15(1):123 – 134, 2007.

[124] Wan-Chun Li, Ping Wei, and Xian-Ci Xiao. An adaptive nonlinear filter of discrete-time system with uncertain covariance using unscented Kalman filter. In *Communications and Information Technology, 2005. ISCIT 2005. IEEE International Symposium on*, volume 2, pages 1436–1439, October 2005.

[125] Kwang Hoon Kim, Jang Gyu Lee, and Chan Gook Park. Adaptive two-stage Kalman filter in the presence of unknown random bias. *International Journal of Adaptive Control and Signal Processing*, 20(7):305–319, 2006.

[126] Kwang Hoon Kim, Jang Gyu Lee, and Chan Gook Park. The stability analysis of the adaptive two-stage Kalman filter. *International Journal of Adaptive Control and Signal Processing*, 21(10):856–870, 2007.

[127] Kwang Hoon Kim, Jang Gyu Lee, Chan Gook Park, and others. Adaptive two-stage EKF for INS-GPS loosely coupled system with unknown fault bias. *Positioning*, 1(10), 2006.

[128] Kwang Hoon Kim, Jang Gyu Lee, and Chan Gook Park. Adaptive two-stage extended Kalman filter for a fault-tolerant INS-GPS loosely coupled system. *Aerospace and Electronic Systems, IEEE Transactions on*, 45(1):125–137, 2009.

[129] Kwang-Hoon Kim, Gyu-In Jee, Chan-Gook Park, and Jang-Gyu Lee. The stability analysis of the adaptive fading extended Kalman filter using the innovation covariance. *International Journal of Control, Automation and Systems*, 7(1):49–56, 2009.

[130] Vahid Fathabadi, Mehdi Shahbazian, Karim Salahshour, and Lotfollah Jargani. Comparison of adaptive Kalman filter methods in state estimation of a nonlinear system using asynchronous measurements. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 2, 2009.

[131] Junchuan Zhou, Stefan Knedlik, and Otmar Loffeld. INS/GPS Tightly-coupled Integration using Adaptive Unscented Particle Filter. *Journal of Navigation*, 63(03):491–511, 2010.

[132] Qi Song, Zhe Jiang, and Jianda Han. Noise Covariance Identification Based Adaptive UKF with Application to Mobile Robot Systems. In *Robotics and*

Automation, 2007 IEEE International Conference on, pages 4164–4169, April 2007.

[133] Qi Song and Yuqing He. Adaptive unscented Kalman filter for estimation of modelling errors for helicopter. In *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, pages 2463–2467, December 2009.

[134] M. Majeed and Indra Narayan Kar. Aerodynamic parameter estimation using adaptive unscented Kalman filter. *Aircraft Engineering and Aerospace Technology*, 85(4):267–279, 2013.

[135] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948 vol.4, November 1995.

[136] Russ C Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science*, volume 1, pages 39–43. New York, NY, 1995.

[137] R.K. Jatoth and T.K. Kumar. Particle Swarm Optimization Based Tuning of Unscented Kalman Filter for Bearings Only Tracking. In *Advances in Recent Technologies in Communication and Computing, 2009. ARTCom '09. International Conference on*, pages 444–448, October 2009.

[138] Jianbin Xin, Guimin Chen, and Yubao Hai. A Particle Swarm Optimizer with Multi-stage Linearly-Decreasing Inertia Weight. In *Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on*, volume 1, pages 505–508, April 2009.

[139] Dah-Jing Jwo and Shun-Chieh Chang. Application of Optimization Technique for GPS Navigation Kalman Filter Adaptation. In De-Shuang Huang, II Wunsch, DonaldC., DanielS. Levine, and Kang-Hyun Jo, editors, *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, volume 5226 of *Lecture Notes in Computer Science*, pages 227–234. Springer Berlin Heidelberg, 2008.

[140] N. Ramakoti, A. Vinay, and R.K. Jatoth. Particle Swarm Optimization Aided Kalman Filter for Object Tracking. In *Advances in Computing, Control, Telecommunication Technologies, 2009. ACT '09. International Conference on*, pages 531–533, December 2009.

[141] Ravi Kumar Jatoth and T KISHORE Kumar. Particle Swarm optimization Based Tuning of Extended Kalman Filter for Manoeuvring Target Tracking. *International Journal of Circuits, Systems and Signal Processing*, 3(3):127–136, 2009.

[142] Ravi Kumar Jatoth and T Kishore Kumar. Swarm intelligence based tuning of unscented Kalman filter for bearings only tracking. *Int. J. of Recent Trends in Engineering and Technology*, 2(5), 2009.

[143] R.K. Jatoth, D.N. Rao, and K.S. Kumar. Particle Swarm Optimization aided unscented kalman filter for ballistic target tracking. In *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, pages 455–460, October 2010.

[144] T. K. Panigrahi, P. K. Dash, and P. K. Hota. A self-tuning optimised unscented Kalman filter for voltage flicker and harmonic estimation. *International Journal of Power and Energy Conversion (IJPEC)*, 2(3), 2010.

[145] Jeff Delaune. Vision-Based Navigation Test Bench: Design and Operation, August 2012.

[146] Thomas Voirin, Jeff Delaune, Guy Le Besnerais, Jean-Loup Farges, Clement Bourdarias, and Hans Krueger. Challenges of Pinpoint Landing for Planetary Exploration: The Lion Absolute Vision-Based Navigation System Step-Wise Validation Approach. San Jose State University, California, USA, June 2013.

[147] DLR. VISILAB Terrain Model and DEM Reference, April 2012.

[148] Jody L Davis, Scott A Striepe, Robert W Maddock, Andrew E Johnson, and Stephen C Paschall. POST2 End-to-End Descent and Landing Simulation for ALHAT Design Analysis Cycle 2. Honolulu, Hawaii, August 2008.

[149] S.C. Paschall, T. Brady, B.E. Cohanim, and R. Sostaric. A Self Contained Method for Safe & Precise Lunar Landing. In *Aerospace Conference, 2008 IEEE*, pages 1–12, March 2008.

[150] Ruo Zhang, Ping-Sing Tsai, J. E. Cryer, and M. Shah. Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706, August 1999.

[151] Katsushi Ikeuchi and Berthold K. P. Horn. *Numerical Shape from Shading and Occluding Boundaries*. 1981.

[152] B. K.P. Horn. SHAPE FROM SHADING: A METHOD FOR OBTAINING THE SHAPE OF A SMOOTH OPAQUE OBJECT FROM ONE VIEW. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1970.

[153] A. P. Pentland. Local Shading Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(2):170–187, March 1984.

[154] A. Pentland. Shape Information From Shading: A Theory About Human Perception. In *[1988 Proceedings] Second International Conference on Computer Vision*, pages 404–413, December 1988.

[155] Tsai Ping-Sing and Mubarak Shah. Shape from shading using linear approximation. *Image and Vision Computing*, 12(8):487 – 498, 1994.

[156] Robert J. Woodham. Shape from shading. pages 513–531. MIT Press, Cambridge, MA, USA, 1989.

[157] Jean-Denis Durou, Maurizio Falcone, and Manuela Sagona. Numerical methods for shape-from-shading: A new survey with benchmarks. *Computer Vision and Image Understanding*, 109(1):22 – 43, 2008.

[158] Clay Matthew Thompson. *Robust Photo-topography by Fusing Shape-from-Shading and Stereo*. PhD thesis, Massachusetts Institute of Technology, 1993.

[159] C. Heipke. Integration of Digital Image Matching and Multi Image Shape from Shading. In S. Fuchs and R. Hoffmann, editors, *Mustererkennung 1992: 14. DAGM-Symposium, Dresden, 14.16. September 1992*, pages 186–198. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992.

[160] Christian Heipke and Christian Piechullek. Toward surface reconstruction using multi-image shape from shading. In *SPIE*, volume 2357, pages 361–369, 1994.

[161] Christian Heipke, Christian Piechullek, and Heinrich Ebner. Simulation studies and practical tests using multi-image shape from shading. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 56(2):139 – 148, 2001.

[162] Volker Lohse and Christian Heipke. Multi-image shape-from-shading: Derivation of planetary digital terrain models using clementine images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 35:828–833, 2004.

[163] Volker Lohse, Christian Heipke, and Randolph L. Kirk. Derivation of planetary topography using multi-image shape-from-shading. *Planetary and Space Science*, 54(7):661 – 674, 2006.

[164] Christian Whler. Shape from Shading Under Coplanar Light Sources. In Carl Edward Rasmussen, Heinrich H. Blthoff, Bernhard Schlkopf, and Martin A. Giese, editors, *Pattern Recognition: 26th DAGM Symposium, Tbingen, Germany, August 30 - September 1, 2004. Proceedings*, pages 278–285. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[165] S. Herbort, A. Grumpe, and C. Whler. Reconstruction of non-Lambertian surfaces by fusion of Shape from Shading and active range scanning. In *2011 18th IEEE International Conference on Image Processing*, pages 17–20, September 2011.

[166] R. OHara and D. Barnes. A new shape from shading technique with application to Mars Express {HRSC} images. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 67:27 – 34, 2012.

[167] J. T. Barron and J. Malik. High-frequency shape and albedo from shading using natural image statistics. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2521–2528, June 2011.

[168] J. T. Barron and J. Malik. Shape, Illumination, and Reflectance from Shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, August 2015.

[169] Stefan Roth and Michael J. Black. Fields of Experts: A Framework for Learning Image Priors. In *In CVPR*, pages 860–867, 2005.

[170] Stefan Roth and Michael J. Black. On the Spatial Statistics of Optical Flow. *International Journal of Computer Vision*, 74(1):33–50, 2007.

[171] Yair Weiss and William T. Freeman. What makes a good model of natural images. In *in: CVPR 2007: Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society*, pages 1–8, 2007.