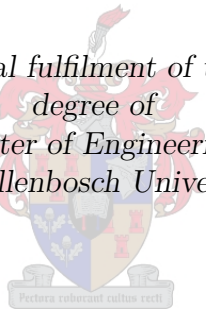
Space Debris:
Pose Estimation Using Stereo Vision

by

Willem Carel de Jongh

Thesis presented in partial fulfilment of the requirements for the degree of
Master of Engineering
at Stellenbosch University



Supervisors:

Dr H.W. Jordaan Dr C.E. van Daalen

Department Electrical and Electronic Engineering

April 2019

Plagiarism Declaration

1. I have read and understand the Stellenbosch University Policy on Plagiarism and the definitions of plagiarism and self-plagiarism contained in the Policy [Plagiarism: The use of the ideas or material of others without acknowledgement, or the re-use of one's own previously evaluated or published material without acknowledgement or indication thereof (self-plagiarism or text-recycling)].

2. I also understand that direct translations are plagiarism.

3. Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.

4. I declare that the work contained in this assignment is my own work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

	W.C. de Jongh	***April 2019***
Student Number	Initials and Surname	Date

Abstract

Tracking the relative attitude and position of uncooperative in-orbit objects is vital for autonomous operations in space. Vision-based solutions have low power consumption and can provide a system with valuable information to perform pose determination. Estimation algorithms are required to extract the system states from visual measurements and many similar approaches have been investigated in mobile robotics.

In this thesis, a chaser satellite is fitted with stereo cameras which are used to extract unique features on the surfaces of an uncooperative, unknown target. The scale invariant feature transform (SIFT) feature detector is used to identify and establish correspondence of the target features. A full state kinematic estimator is implemented using an extended Kalman filter (EKF) based on the simultaneous localisation and mapping (SLAM) approach. The filter makes use of the observations from the feature extractor to estimate the position and orientation of the target relative to the chaser along with the angular and linear velocities of the target. Shape and size reconstruction of the target is made possible using the sparsely tracked features.

A simulation environment providing ground truth is used to verify the performance of the estimation algorithm. The integration of the estimator with the feature extractor is assessed using real world data. Experimental data is obtained from image sequences of a moving target in a laboratory set-up. Results show that the filter estimates the system states successfully and that the developed feature extractor is capable of detecting robust features with reliable correspondence. It is concluded that the use of a stereo-vision-based estimator is a viable option for autonomous operations such as space debris removal and satellite service missions.

Uittreksel

Akkurate lokalisering van onbekende ruimte voorwerpe in verhouding tot 'n volgersatelliet is noodsaaklik vir outonome ruimte operasies. Skatting van die oriëntasie en posisie van die voorwerp deur middel van visuele sensors soos kameras, is 'n gewilde oplossing in die robotika-veld. Visuele sensors het 'n lae kragverbruik en is goedkoop om te implementeer. Lokaliseringsalgoritmes word benodig om die toestande van die voorwerp uit die visuele metings te onttrek.

Hierdie tesis bespreek 'n stereo-kamera paar wat, saam met die skaal bestande kenmerk transform (SIFT) algoritme, gebruik word om unieke punte op die oppervlaktes van 'n nie-samewerkende, onbekende voorwerp te vind. Die algoritme is só ontwerp om rekord te hou van ooreenstemmende punte in opeenvolgende beelde. 'n Kinematiese toestands-skatter word geïmplementeer met behulp van 'n uitgebreide Kalman filter (EKF). Die skattingsalgoritme gebruik die gelyktydige lokalisering en kartering (SLAM) benadering. Die filter skat die relatiewe posisie en oriëntasie van die voorwerp af met betrekking tot die kameras. Die hoek- en lineêre-snelhede van die voorwerp word ook onttrek. 'n Verteenwoordiging van die voorwerp se grootte en vorm word saamgestel vanuit die geskatte posisies van die unieke voorwerp-punte.

'n Simulasie-omgewing, wat grondwaarheid voorsien, word gebruik om die werking van die skattingsalgoritme te toets. Die integrasie van die skatter met die beeldverwerkingsalgoritme word getoets deur gebruik te maak van eksperimentele beelde. Eksperimentele beelde word vasgelê deur 'n bewegende voorwerp waar te neem in 'n laboratorium-opstelling.

Die stelsel toon bevredigende uitslae. Die EKF-SLAM benadering, in samewerking met die beeldverwerkingsalgoritme, is daartoe in staat om die voorwerp te lokaliseer relatief tot die kameras. Die studie kom tot die gevolgtrekking dat stereo-visiegebaseerde skatters voldoende is vir outonome ruimtesendings soos die diens van satelliete en die verwydering van ruimterommel.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations and Acronyms

ADR	active debris removal
CAD	computer aided design
CRF	camera reference frame
DCM	direction cosine matrix
DEOS	Deutsche Orbitale Servicing Mission
EKF	extended Kalman filter
FOV	field of view
ICP	iterative closest point
ISS	International Space Station
KF	Kalman filter
LEO	low earth orbit
Lidar	light detection and ranging
RMS	root mean square
ROI	region of interest
SFM	structure from motion
SLAM	simultaneous localisation and mapping
SIFT	scale invariant feature transform
SURF	speeded up robust features
UKF	unscented Kalman filter

Syntax and Style

x	The scalar value x
$\mathbf{x}$	The vector $\mathbf{x}$ (usually lowercase)
$\bar{\mathbf{x}}$	Unit vector
X	The matrix X (usually uppercase)
$\mathbf{A}_{\mathcal{X}}^{\mathcal{Y}}$	The attitude of $\mathcal{Y}$ relative to $\mathcal{X}$

Chapter 1

Introduction and Problem Description

1.1 Problem Background

Over the past decade it has become evident that the success of current and future space missions is jeopardised by the presence of space debris. More than 8500 spacecraft launches have taken place since the launch of the Soviet Union satellite, Sputnik1 in 1957 [1]. A large number of man-made debris has been created by these space activities. Decommissioned satellites, rocket bodies and other hardware are scattered around earth in various orbits. These objects have very high velocities and a collision with a satellite or spacecraft can lead to immediate mission failure.

Presently, passive debris removal systems are incorporated in new space missions to reduce the number of new debris caused by the mission. This includes drag sails and inflatable balloons that deploy at end-of-life to force the satellite to de-orbit. American aerospace company, Space-X, has also recently implemented reusable heavy-lift engines like the Falcon 9 [2], which reduces the amount of debris caused by satellite launches. Passive methods, however, will not aid in reducing the estimated 500 000 pieces of orbital debris that already exist [3]. Active debris removal (ADR) methods and missions are required to decrease the amount of debris and consequently mitigate the risk of collisions.

Accurate pose information of a target relative to a chaser satellite is required in order for a chaser satellite to perform ADR to capture and remove the target from its orbit. The appearance and shapes of debris pieces are generally unknown to the chaser before the mission starts, thus no prior information can be utilised. The target in the case of this project is assumed to be uncooperative, meaning that there is no communication link between the chaser and the target. A close-range measurement system is therefore required to determine the pose of the target and its angular and linear velocities relative to the chaser. This information is critical to perform safe rendezvous operations with the target.

1.2 Project Aim and Proposed Solution

This project aims to develop a system capable of determining the relative pose between an unknown, uncooperative target and a chaser satellite. A stereo-camera sensor is used to perform visual-based estimation of a freely moving target. Measurements are based on the calculation of geometric positions of natural features on the target. A tracking algorithm is implemented to estimate the state of the target's pose along with the positions of the features on the target.

1.3 Document Outline

The next chapter investigates aspects of pose determination for space based applications. The current state and challenges of space debris and ADR is discussed. Chapter 3 defines the project problem in detail and describes the system model. The relevant reference frames used in this project are also introduced. Sensor development is discussed in Chapter 4. This deals with modelling the sensor, developing a measurement extraction algorithm and describing the associated image processing algorithms. Chapter 5 discusses the development of the tracking algorithm and provides results that were obtained from simulated measurements. A practical experiment was conducted and the results are presented in Chapter 6. Final conclusions, findings and future recommendations are drawn in Chapter 7.

Chapter 2

Literature Study

2.1 Introduction

This chapter investigates typical space-based pose determination applications and how they can be exploited for debris removal. Aspects of space debris are explored to gain a better understanding of why and how to mitigate debris. Two recent ADR missions are investigated to identify the current state-of-the-art technologies that are implemented. Several methods for ADR are also discussed along with their associated risks and the sensor information required to successfully execute these missions. Previously researched methods for determining the pose of in-orbit objects are reviewed and conclusions are made with regards to the effectiveness of these methods and their relevance to this project.

2.2 Space-Based Pose Determination

Pose determination is concerned with the calculation of the relative position and attitude of a target object in orbit relative to a chaser satellite. The application is specifically focused on close-proximity operations like rendezvous, docking, monitoring and servicing [4]. Close-proximity operations require autonomy since delays from ground control communications would deem it impossible to safely perform these procedures. Pose determination is a problem with a broad range of applications, as shown in Figure 2.1. Two main categories are defined, namely cooperative and uncooperative pose determination. A target is considered to be cooperative if it possesses useful information that can be exploited for pose estimation, such as on-board sensors and communication links with the chaser.

Uncooperative pose determination is sub-divided into known- and unknown-target applications. Prior data is available when interacting with known targets. This includes information like geometrical shape and size or the positions of known features like antennas or solar panels. In other cases an exact model of the target is available prior to the mission start [4]. Applications for uncooperative, but known, targets include the removal or servicing of defunct and decommissioned satellites.

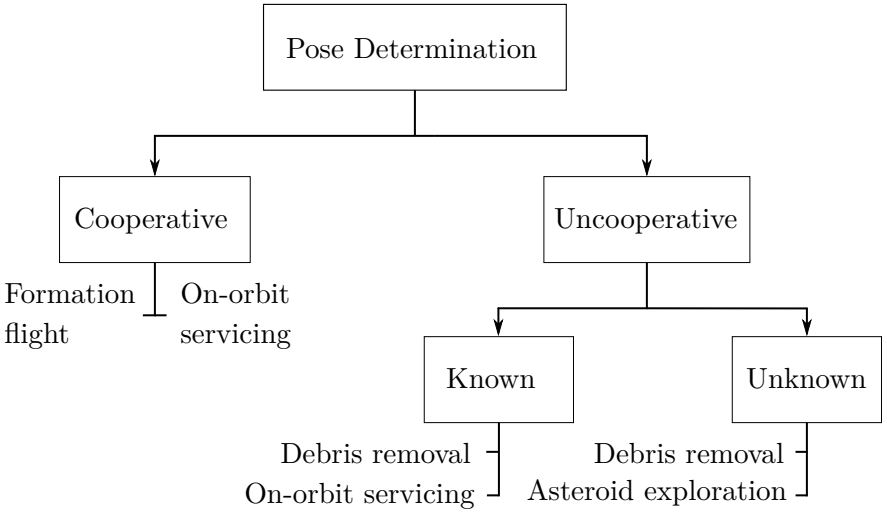


Figure 2.1 – Space-based applications of pose determination.

Applications with no prior target information are classified as missions with uncooperative, unknown targets. Both natural debris such as asteroids and man-made objects like rocket bodies or satellites present the greatest challenges for close-proximity operations since the debris may be tumbling with large angular rates. Models of man-made objects may be inaccurate after years of exposure in space or if the target satellite was damaged to such an extent that it is considered to be fragmentation debris. There are high risks associated with the interaction of uncooperative, unknown targets and therefore the requirements for accuracy and robustness of the pose determination are strict.

Opromolla *et al.* [4] reviewed several techniques for close proximity operations, including typical sensor configurations that are appropriate for the applications shown in Figure 2.1. Relative navigation in space makes use of primarily infrared (active) and visible light (passive) sensors. Laser range finders operate in the infrared spectrum and are capable of measuring large distances with high accuracy. Line-of-sight measurement systems like range finders are, however, not suited for pose determination applications since they make use of non-steerable single lasers, resulting in coarse measurements. Light detection and ranging (Lidar) systems operate in the same frequency spectrum as range finders, but are capable of creating 3D point clouds of scanned objects. Multiple samples of a moving target using Lidar can provide enough information to perform pose estimation of the target [4].

Monocular- and stereo-cameras are deemed passive sensors operating in the visible light spectrum. Cameras have a lower power consumption than active sensors and are less expensive to implement than Lidar. An additional benefit of camera sensors lies in the fact that they are easier to use for human-in-the-loop actions. This includes ground control verifications, human controlled operation and visual confirmation. Cameras do however, suffer in poor lighting conditions or when segmentation errors occur and objects in the background are tracked with the target. The former can be overcome by fitting the chaser with illumination devices like the

system used by the Proximity Operation Sensor (PXS). The PXS was developed by the National Space Agency of Japan (NASDA) [5]. The camera sensor was equipped with a light emitting diode (LED) array that emitted visible light in a 30° cone around the sensor.

Stereo cameras are capable of acquiring sparse 3D point cloud information similar to Lidar, although it requires a bit more processing. Data generated by camera sensors are more textured than Lidar scans. Lidar is more accurate over larger distances and the functionality and range of stereo cameras are often limited by the baseline distance between the left and right hand camera.

With all this considered, it is noted that the sensor configuration should be selected based on the mission specifications. Some missions choose to make use of several different sensors depending on the mission phase [6; 7]. Therefore, the optimal solution may lie in the combination of sensors working in different frequency and light domains, using data fusion to perform the best possible pose estimation given the problem.

2.3 Overview of Space Debris

Kessler and Cour-Palais [8] predicted in 1978 that the frequency of collisions between satellites would increase as more objects are launched into space. The density of objects in low earth orbit (LEO) would become so high that an ablation cascade will occur where the exponential increase in collisions will render space inaccessible for many years.

The U.S. Space Surveillance Network maintains a catalogue of space debris objects. The data shown in Figure 2.2 shows two significant increases in the amount of debris in 2007 and 2009. The first increase in 2007, was caused by an anti-satellite missile test that was conducted by China. A SC-19 missile was used to destroy an ageing Chinese weather satellite [9]. This explosion created more than 3000 pieces of trackable debris. The second was a catastrophic collision between a decommissioned Russian satellite, Kosmos-2251 and a U.S. communication satellite, Iridium 33 in 2009. The collision created over a 1000 pieces of debris [10]. This was the first time in history that an unplanned collision between two satellites occurred, providing a glimpse of the danger of space debris that Kessler and Cour-Palais [8] predicted in 1978.

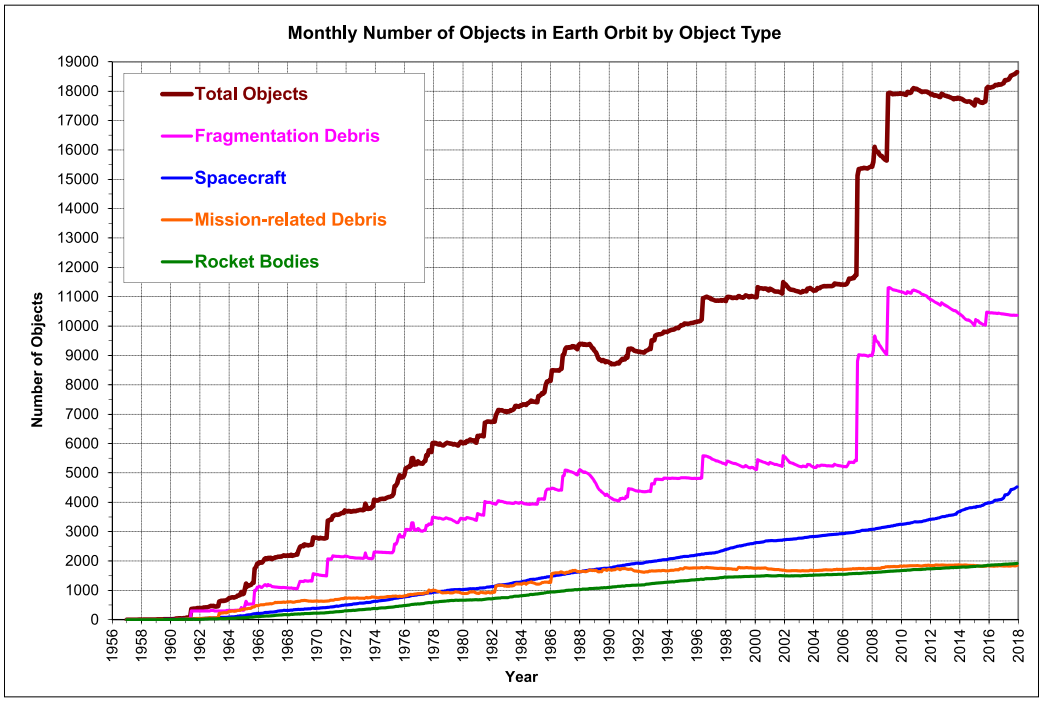


Figure 2.2 – Chart summarising all catalogued objects in orbit around the earth. Catalogue created by the U.S. Space Surveillance Network. The pink data represents fragmentation debris, which includes satellite breakup debris, while the mission-related debris in orange includes all dispensed, separated and released objects as part of a mission procedure. Figure reprinted from Orbital Debris Quarterly News, [11].

The Inter-Agency Space Debris Coordination Committee (IDAC) requires that all new satellites that are launched into LEO are equipped with passive removal technologies, which forces the satellite to de-orbit itself within 25 years after its mission ends. Although effective in mitigating the creation of new debris, Liou [12] has found that the amount of debris that already exists will spiral out of control within the next 200 years due to debris inter-collisions. Therefore, the need for ADR is clear, with several methods having been proposed in the past. A study by NASA shows that approximately five objects with high collision probabilities need to be removed per year to prevent the Kessler syndrome [12]. These are objects with masses larger than one ton and with dimensions bigger than two metres in diameter. Potential targets that fit into this category are the upper stages of rocket bodies commonly used to deploy satellites [12].

2.4 Active Debris Removal

Current ADR technologies have limitations. DEOS and RemoveDebris are two ADR missions that aim to investigate a number of debris removal methods.

2.4.1 DEOS

The Deutsche Orbitale Servicing Mission (DEOS) [6] was a German in-orbit satellite servicing concept with the purpose to evaluate technology for rendezvous, capture and de-orbiting of uncooperative satellites. The proposed mission consisted of two satellites. A client satellite serves as the non-cooperative target and is captured by the active chaser satellite. The chaser satellite is equipped with a robotic manipulator for berthing with the target. The objectives of the mission, illustrated in Figure 2.3, were:

- Use a robotic manipulator to de-tumble and capture a non-cooperative target.
- Simulate a servicing mission such as refuelling or repairing the target.
- Remove the target from orbit by forcing it to re-enter the atmosphere.

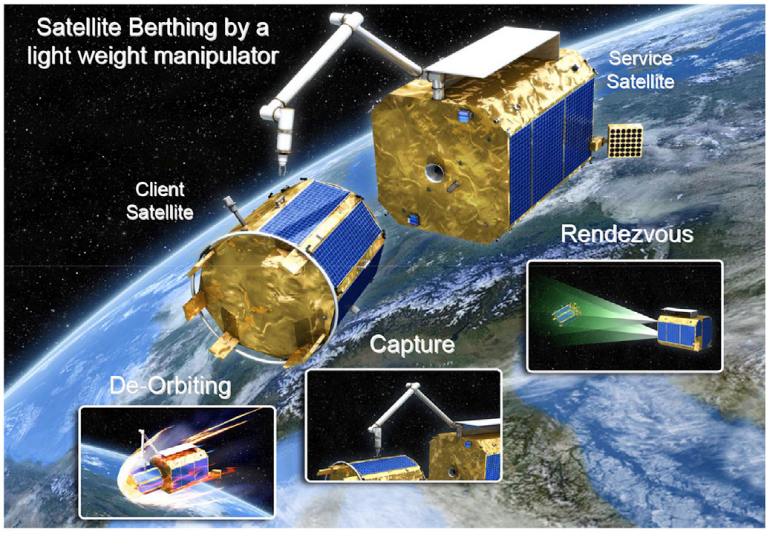


Figure 2.3 – Illustration of the DEOS mission objectives. Reprinted from Reintsema *et al.* [6].

The chaser satellite is equipped with several sensors, each designed for a different phase of the mission. These include a far-range monocular camera (FMC), Lidar, Radar, a mid-range stereo camera (MSC), a close-range stereo camera (CSC) and a monocular docking camera (MDC). Rendezvous with the target consists of four phases, each with their respective sensors as shown in Table 2.1. An inspection manoeuvre is done to estimate the motion of the target. The primary sensors used for pose determination are the mid-range stereo cameras and radar.

Phase	Range	Sensors	Description
Formation Flying and Phasing	>4 km	FMC	Homing in on target using absolute navigation techniques. Reduce phase angles between target and chaser.
Far Range Approach	85 m - 4 km	FMC Lidar	Proceed to close in on target and switch over to relative navigation techniques.
Mid Range Approach	6 m - 85 m	RADAR Lidar MSC	Final approach of target. If target is unknown or non-cooperative, execute a fly-around inspection of the target at 15 m.
Close Proximity Operation	<6 m	RADAR MDC	Follow a straight line trajectory to the target and proceed with berthing operations.

Table 2.1 – DEOS sensor configuration.

Reintsema *et al.* [6] also investigated the effect of illumination changes on the performance of the camera sensors. It was found that additional lighting is required when difficult illumination conditions arise, such as total darkness or if the target is positioned between the sun and the chaser. The project timeline proposed the launch of the mission in 2018, but the project was cancelled soon after the definition phase.

2.4.2 RemoveDebris

The RemoveDebris mission was launched in 2017 with the aim of performing a low-cost ADR demonstration using a net and harpoon [13]. Vision based navigation (VBN) and drag sails are also implemented to test the feasibility of these technologies in a space environment. Two debris satellites, DS-1 and DS-2 shown in Figure 2.4, are loaded into the main mission platform. These are used as target debris and are launched from the platform after deployment into orbit from the International Space Station (ISS). The main platform was successfully released from the ISS on the 20^{th} of June 2018 [14].

Three key objectives were identified for RemoveDebris over the course of the 6 month operation time [7]:

- Deploy DS-1 at a low velocity ($\approx$ 5 cm/s) from the main platform. After successful deployment, DS-1 inflates a balloon to increase its target area. The

main platform then proceeds to eject a net to capture DS-1. Thereafter, DS-1 is allowed to de-orbit at an accelerated rate due to the increased surface area from the inflated balloon.

- Extend a 10 x 10 cm target 1.5 m away from the platform and strike it with a harpoon. The main platform is equipped with two supervision cameras to assess the outcome of the harpoon and net experiments.
- Deploy DS-2 at a velocity of $\approx$ 2 cm/s from the main platform and perform a series of manoeuvres to capture image and Lidar data of DS-2's movement.

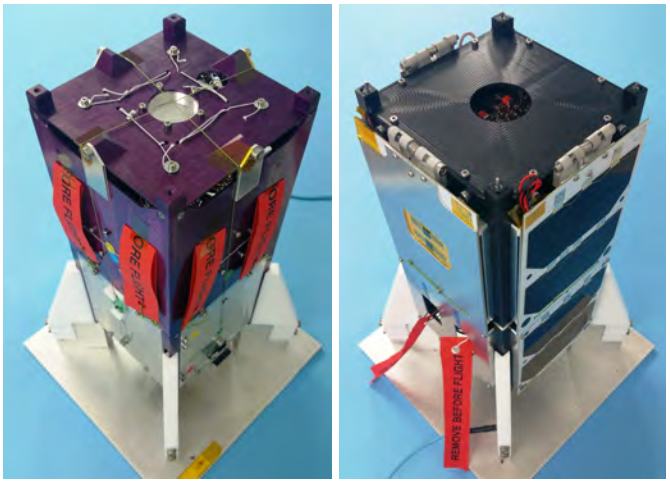


Figure 2.4 – Targets for RemoveDebris mission. DS-1 CubeSat (left) is used for the net experiment and DS-2 CubeSat (right) used for VBN [13].

DS-2 is fitted with deployable panels to represent a satellite object with solar panels. It is also equipped with on-board sensors to determine its own attitude and a data link to the main platform to transfer this ground truth data. The captured data is processed offline using dedicated image processing algorithms. An extended Kalman filter (EKF) performs data fusion and estimates the motion of DS-2. The quality of the estimates is compared to the data that is acquired from DS-2's on-board sensors. RemoveDebris completed the net capture procedure in September of 2018 and the VBN experiment would have commenced in October 2018 [14].

A major factor in the success of DEOS and RemoveDebris is the effectiveness and quality of their respective pose determination systems. In the case of DEOS, failure to predict the target's pose may result in a crash with the target. Although not so critical for RemoveDebris, since a berthing operation is not done, the VBN system plays a vital role in the research of future ADR missions. DEOS makes use of a complex sensor configuration to increase the robustness of the system should one of the sensors fail. However, no consideration was made for additional light or illumination sources, even though it was a concern for the accuracy of the on-board optical sensors. The use of Lidar (DEOS and RemoveDebris) and Radar (DEOS)

overcomes this challenge, but active sensors like Lidar and Radar are expensive and consume more energy than passive optical sensors.

2.4.3 Active Methods for Debris Removal

An ADR mission consists of various phases, as mentioned in Section 2.4.1. In general, the process consists of: launch and early orbit phase (LEOP), far-range rendezvous phase, close-range rendezvous phase, capturing and removal phase [15]. Pose estimation plays a major role in the close-range rendezvous and capturing phases. Hakima and Emami [16] assessed five different ADR methods for LEO debris. Removal by net, laser, electrodynamic tether, ion beam shepherds and robotic arm were selected and their performances compared using a Monte Carlo simulation. The feasibility of these methods were assessed based on several criteria including missions costs, technology readiness and risks associated with each method. Table 2.2 summarises the risks associated with each of these methods.

It was noted that one of the main concerns of contact ADR methods is uncontrolled debris attitude rates and the need for a robust vision/tracking system is evident. Hakima and Emami [16] concluded that throw nets, space-based lasers and robotic arms are the most feasible for ADR given the current state of knowledge and technology readiness.

Method	Mission Risk Description
NET	$\mathcal{R}1$. Explosion of debris energy stores and further fragmentation $\mathcal{R}2$. Damages to remover spacecraft due to uncontrolled debris attitude rates $\mathcal{R}3$. Debris capturing is unsuccessful
LSR	$\mathcal{R}1$. Thrust degradation due to attenuation of the laser beam cause by the ejected plume $\mathcal{R}2$. Explosion of debris energy stores and further fragmentation due to extremely high temperature created during ablation $\mathcal{R}3$. Vision system fails to detect, acquire and track the target debris
IBS	$\mathcal{R}1$. Shepherd fails to keep a safe distance from the target and collides with it $\mathcal{R}2$. Fragmentation of target due to excessive spin-up resulting from misalignment between ion beam centre of pressure and target centre of mass $\mathcal{R}3$. Secondary ions backscattered from the target surface contaminate sensitive parts
EDT	$\mathcal{R}1$. Explosion of debris energy stores and further fragmentation $\mathcal{R}2$. Tether gets tangled or collides with other in-orbit objects and creates more debris $\mathcal{R}3$. Debris capturing is unsuccessful
ARM	$\mathcal{R}1$. Explosion of debris energy stores and further fragmentation $\mathcal{R}2$. Damages to remover spacecraft due to uncontrolled debris attitude rates $\mathcal{R}3$. Debris capturing is unsuccessful

Table 2.2 – Allocated risk attributes of different ADR methods. The ADR methods that are assessed are throw nets (NET), space based lasers (LSR), ion beam shepherds (IBS), electrodynamic tethers (EDT) and capturing via robotic arm (ARM). Adapted from Hakima and Emami [16].

2.5 Pose Estimation Methods From Previous Research

A number of different pose estimation methods, for space-based applications, have been proposed in the past.

Malan [17] used monocular vision to perform relative pose estimation between two satellites in formation flight. This was done by placing fiducials on the target satellite and tracking their positions using the camera sensor. Kalman filters were used to estimate the target pose using known dynamics of the target, including the moments of inertia. Malan was able to successfully and accurately track the target satellite using both simulated and experimental data. Experimental results using real world data with ground truth was acquired by fixing a target to a robotic arm. This method proved to be an effective solution in the case where the shape and mass of the target are known.

Song and Cao [18] also used monocular vision to perform pose estimation of a non-cooperative target. A sliding window Hough transform was used to identify the triangular structures, shown in Figure 2.5, that are commonly used to fix solar panels to satellites. Pose calculations were based on a feature recognition algorithm that compared the image features of the solar panel structures to a pre-loaded model of the target. This method does not require the placement of fiducials or markers, but still relies on the target having triangular solar panel mounts. It was found to be highly accurate, but this approach is not feasible when designing for the capture of fragmentation debris or satellites with a different attachment of solar panels.

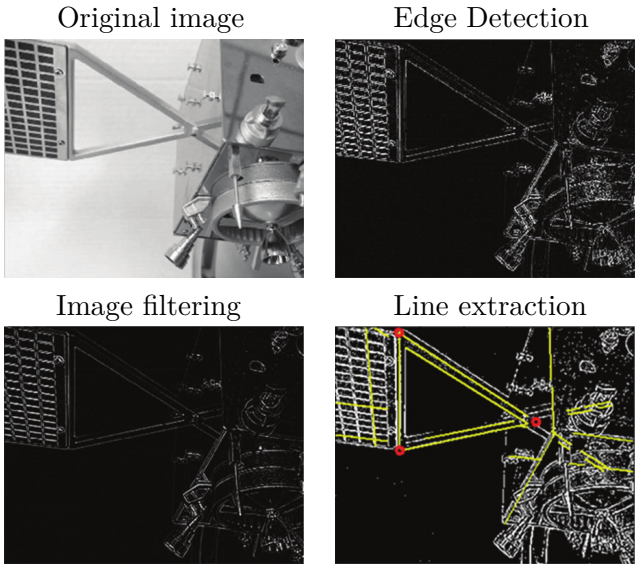


Figure 2.5 – Extracting triangle lines from images of a non-cooperative satellite [18].

Modelling the appearance of a satellite at close range for pose estimation with the application of on-orbit servicing was investigated by Oumer *et al.* [19]. A German satellite, TerraSAR-X, shown in Figure 2.6, was identified as the target for their research. An exact replica of the outer surface of the rear side of the satellite was constructed and multiple images were captured in a spherical arrangement around the target. This learning data was used offline to build a vocabulary tree of image features. Online estimation of the target was performed by retrieving image

correspondences between sensor measurements and the appearance model. The correspondence between 2D image plane features and 3D model features were used with the random sample consensus (RANSAC) algorithm to compute the pose of TerraSAR-X relative to the monocular camera sensor.

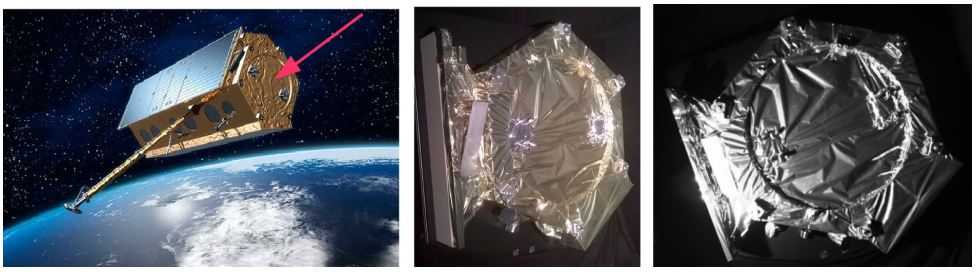


Figure 2.6 – Artistic rendition of TerraSAR-X (left) and the rear side replica of the satellite (right), used for appearance learning [19].

Dong [20] proposed the dynamic capturing of a non-cooperative object by means of a robotic arm. The pose of the target was determined using a vision-based tracking system that used an EKF to perform recursive estimation of the target's states. A monocular camera was attached to the end of the robot to capture images of the moving target. However, the target was only moving in one plane and therefore depth calculation was not necessary.

Lichter and Dubowsky [21] proposed an architecture to compute the states and shape of an object using 3D vision sensors. Their solution overcomes the drawbacks of the methods proposed by Malan [17] and Song and Cao [18] by making use of four 3D sensors arranged in a tetrahedron formation around the target as shown in Figure 2.7. A cloud of points is constructed at every timestep using the data captured by the sensors. A coarse estimate of the target's pose is computed by means of an eigenvalue decomposition using the relative geometric moment of inertia tensor. The inertia tensor is calculated using the point cloud information. The coarse pose estimates are fed to an EKF that makes use of a dynamic motion model to provide a refined pose estimate along with a probabilistic reconstruction of the target. Although this solution provides accurate estimates of the target's states and shape in simulation, the use of 3D sensors on multiple satellites is not practical. This would require the four separate satellites with their own navigation systems to continuously update their positions relative to each other to achieve accurate capturing of the point cloud.

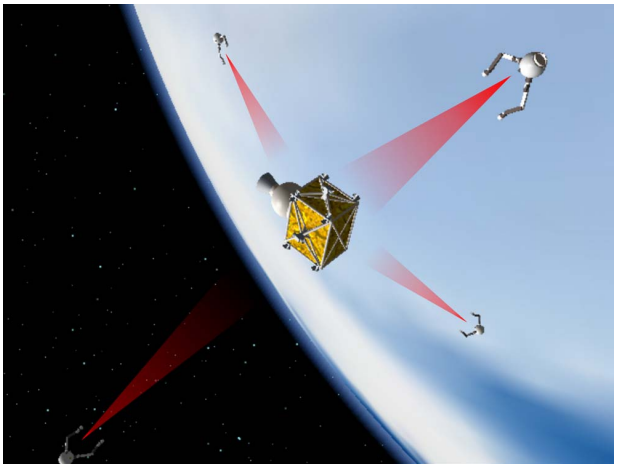


Figure 2.7 – Cooperative sensor configuration as proposed by Lichter and Dubowsky [21]. Four 3D sensors are arranged around the target to capture a dense point cloud of the target.

He *et al.* [22] proposed a solution similar to Lichter and Dubowsky [21]. Non-cooperative pose estimation is performed using point cloud data generated by a series of Lidar images. A particle filter uses the curvature, densities and geometric characteristics of the points to estimate the pose of the unknown moving object. Due to the availability of only sparse point cloud data, He *et al.* opted to rather use a kinematic motion model. Although it may yield slower convergence or erratic behaviour, the use of a kinematic model makes much more sense, since inertia and mass information is normally not available when dealing with space debris. It was, however, concluded that the use of the particle filter is computationally expensive and an alternative solution or improvement in the algorithm architecture is required for real-time pose estimation applications.

The iterative closest point (ICP) algorithm is often used for pose estimation by searching for the best alignment between two point clouds. The distance between corresponding points is iteratively minimized until a best fit is found. Shahid and Okouneva [23] investigated pose determination of high value satellites in geo-synchronous orbits using Lidar scans and ICP matching. Scans of satellite surfaces were matched with the known computer aided design (CAD) models of the targets. Woods and Christian [24] also used the ICP algorithm with flash Lidar measurements. It was found that even though the ICP algorithm is able to converge quickly, around 25 iterations in the case of Shahid and Okouneva [23], that a good initial guess is required. A poor initial guess can lead to a significant errors in pose estimation as illustrated in Figure 2.8. The shortcomings with using the ICP algorithm alone was overcome by Woods and Christian [24] by implementing a novel clustered viewpoint feature histogram (CVFH) method. This solution was coupled with an EKF to perform pose estimation of the target.

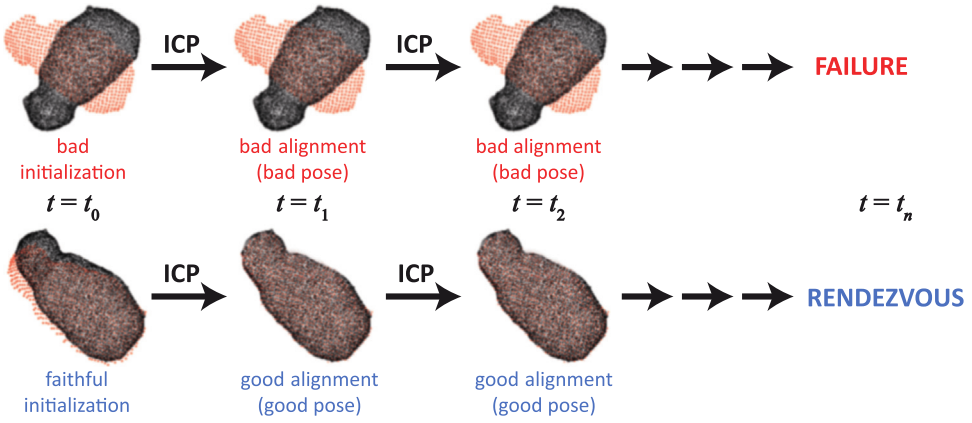


Figure 2.8 – Illustration of the influence of poor a initial guess on the performance of the ICP algorithm. Wrong convergence to a local minimum due to errors in the initial guess is shown in the top part of the figure. The bottom part shows correct convergence to the absolute minimum point and the achievement of successful pose matching. Figure reprinted from Woods and Christian [24].

A novel pose estimation method was proposed by Pesce *et al.* [25] using a stereo vision sensor and an EKF. No prior information regarding the target was assumed, but relative attitude dynamics was still exploited by performing a parametrisation of the target's inertia matrix. Results from this research showed that inertia components can successfully be estimated, given that the target has high angular velocities. This helps the dynamic model to converge to consistent and exact values for the inertia parameters. Incorrect inertia ratios resulted in a decrease in the capability to accurately determine the angular velocities and the attitude vectors. Pesce *et al.* [25] also concluded that a robust and reliable stereo feature tracker is required for space-based applications. Occlusions and adverse lighting can have a significant influence on the capabilities of the vision system and the method heavily relies on accurate point cloud information of the target [25].

Biondi *et al.* [26] proposed a method for obtaining space debris angular rates using feature-based Kalman filtering techniques. This method provides a solution to the limitations of Segal *et al.* [27] and Pesce *et al.* [25] where the estimation methods diverge when a lack of consecutive measurements exists. When this is the case, no measurement updates can be done by the Kalman filter and the estimation is purely dependent on the prediction made by the motion model of the target. Stereo vision sensors are used to track target features and a partial pose is computed at timesteps where features are indeed available. The complete attitude signal is then retrieved using compressed sensing techniques often used for signal recovery. Non-linear programming is used based on the assumption that the signal can be represented as a linear combination of a few independent non-linear basis functions [26]. A linear Kalman filter is used to estimate the angular velocities of the target once a full attitude quaternion is available.

A stereo-vision-based filter was developed by Segal *et al.* [27] to estimate the state of a target relative to a chaser satellite. A coupled dynamic motion model was used to

predict the state propagation. Multiple iterated extended Kalman filters (IEKF's) were run in parallel to select an inertia tensor that best suited the dynamic behaviour of the observed object. Analysis indicated that attitude and structure information could be extracted from spatial measurements and could be used for estimating a the full state of a target relative to the sensors.

A survey was done by Chen [28] to investigate the present state and use of Kalman filters for robotic vision and perception applications. It was found that 800 publications were made between 1983 and 2010 where some form of the Kalman filter was used for vision-based implementations. More than 20 variants of the Kalman filter (KF) have been developed over the past 30 years. The main consideration in the complexity of the algorithms lies within the linearity of the systems. Several solutions are used to efficiently deal with non-linear systems, with the EKF and unscented Kalman filter (UKF) being the most common variants. The EKF linearises the system states around the current mean and covariances, where the UKF chooses sample points from the current state distribution, puts them through the non-linear function and creates a new distribution based on the transformed points. It was found that more than 70% of the recent publications in the field of robotic vision used the EKF for non-linear state estimation [28]. Noteworthy cases where the EKF is used to track objects for space based applications are found in the works of Malan [17]; Lichter and Dubowsky [21]; Dong and Zhu [20] and Segal *et al.* [27].

2.6 Conclusion

The current state of space debris and ADR methods has been investigated. Space debris is a real threat to the viability of space exploration in the future. Active debris removal is a heavily researched field and a lot of progress has been made towards it. RemoveDebris was able to successfully capture debris using its net experiment in September of 2018 [14]. This is the first step to a cleaner space, but many challenges still exist. Uncooperative objects with unknown physical properties poses many technical difficulties and not one method is suitable to deal with all types of scenarios [15].

Obtaining accurate pose information of debris and uncooperative objects still remains a difficult task and selecting the correct sensors and estimation algorithms is not an trivial undertaking. Pose estimation using spatial sensors is a well-known field and combining multiple sensors is a popular way to improve system reliability [6; 28]. The use of a monocular camera as primary sensor has been suggested by Malan [17], Song and Cao [18], Oumer *et al.* [19] and Dong and Zhu [20]. Monocular vision is elegant and is the least expensive method to implement in terms of hardware and computation costs. On its own, however, it is not suitable for uncooperative, unknown pose determination since depth information cannot be calculated. Auxiliary sensors like Lidar or range finders are required to make this a feasible option.

The use of stereo camera sensors was explored by Pesce *et al.* [25], Biondi *et al.* [26] and Segal *et al.* [27]. Stereo vision cameras overcome the drawback of solely using monocular cameras, since triangulation is done using point correspondences.

This makes stereo cameras viable as a standalone sensor for pose determination of unknown targets.

It was found that passive sensors like cameras experience a decrease in performance when operating in bad lighting conditions. Maintaining correspondence is also a challenge when using visible light sensors. The option of using Lidar for pose estimation was also suggested in literature [21; 22; 23; 24] and it can be concluded that Lidar sensors can deliver reliable measurements over large distances, but require more processing and have higher implementation costs. Lidar sensors are usually used with an ICP algorithm to perform model matching of the target.

Kalman filters were found to be used extensively throughout this study. The EKF is the most popular filter used in the past 30 years for pose estimation in the field of autonomous robotics [28]. The performance of Kalman filters is, however, dependent on the input of initial conditions and measurements with strong assumptions about noise properties. This does not affect the convergence quality, but may influence the performance of the filter in terms of convergence speeds and accuracy [20].

It is evident from this investigation that pose estimation is a genuine problem with real-world applications in various fields. The chapters that follow will present a solution to determine the position and attitude of an uncooperative, unknown target relative to a sensor.

Chapter 3

Modelling

3.1 Introduction

This project focuses on the pose estimation of a rigid body relative to an observer. This is essentially a localisation and tracking problem and requires a realistic description of the system. The aim of this chapter is to sufficiently define the problem and the proposed solution. Estimation algorithms are discussed and an estimator is chosen to solve the localisation problem. Further, attitude representations of a rigid body are introduced along with the dynamic and kinematic models used to describe the motion of a freely rotating target in inertial space. Attention is given to quaternion attitude representations along with their propagation using angular rates.

3.2 Problem Definition

The mechanics for the rendezvous phase of active debris removal (ADR) are described by different parts, namely chaser attitude dynamics and kinematics, target attitude dynamics and kinematics and the relative translational dynamics and kinematics between the target and the chaser. In this project, it is assumed that the chaser's attitude and position is known and that the inertial reference frame is fixed to the chaser's reference frame, $(\mathcal{C})$. Therefore only the target's translation and attitude mechanics are modelled and described relative to the chaser.

The problem of localising a robot in an unknown environment is often solved using simultaneous localisation and mapping (SLAM). SLAM is a method used by a robot to map an unknown environment and simultaneously locate itself in the map [29]. The probabilistic SLAM approach is used to estimate the pose of a sensor/robot relative to its environment [30]. The sensor, with a pose, $\mathbf{x}_t$, receives control inputs, $\mathbf{u}_t$, and measurements, $\mathbf{z}_t$, at a given timestep, t. Given the inputs and measurements, the aim is to estimate the location of the landmarks and the sensor's location relative to the landmarks. The sensor, with reference frame $\mathcal{C}$, shown in Figure 3.1, moves in an environment and uses landmarks in its vicinity to estimate its attitude, position and velocity relative to the inertially defined reference frame, $\mathcal{I}$. In this case

the inertial reference frame is fixed to the map, $\mathbf{m}$, which consists of the landmark locations.

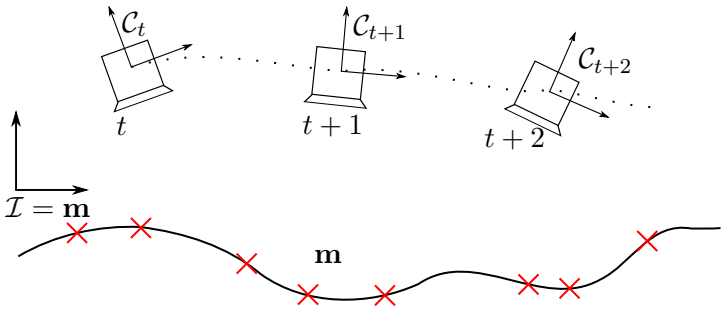


Figure 3.1 – Illustration of the classic SLAM problem, where a robot moves through an unknown environment. Landmarks, indicated by the red crosses, are observed and used to build a map $\mathbf{m}$, of the environment and is used by the robot to localise itself in the map.

For this project, the sensor is considered stationary and the environment or target, moves relative to the sensor. The sensor reference frame is fixed to a chaser satellite and the target can be a debris object or another satellite. Instead of using landmarks like trees or buildings like in the typical SLAM problem, distinct reference points on the target are used. Figure 3.2 illustrates a target with a body-fixed reference frame ($\mathcal{B}$), moving in inertial space fixed to the sensor reference frame. Features on the target are denoted by red crosses and are used to build a body-fixed map ($\mathbf{m}$) of the target and simultaneously estimate the pose of the target relative to the sensor. There are no control inputs, $\mathbf{u}_t$, in this problem, since the sensor is stationary and the target is uncooperative.

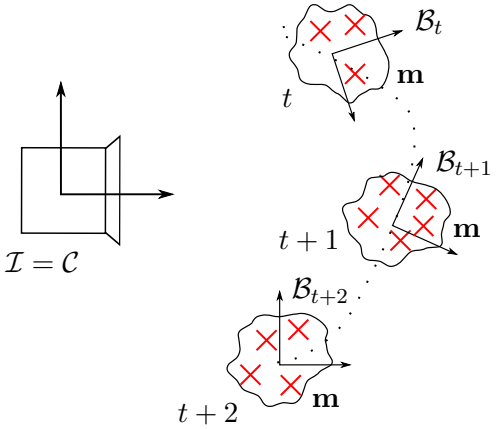


Figure 3.2 – A stationary sensor or robot observes a moving target in inertial space. The features on the target are indicated by red crosses and are used to construct a map of features on the target over time. The movement of these features are used to estimate the motion of the target relative to the sensor.

The estimation solution is composed of two distinct parts that are designed separately. The first is a stereo camera sensor that provides 3D measurements of the target. Secondly, the observations are used in an estimator algorithm that extracts the states using kinematic models of the system. A block diagram of the system framework is shown in Figure 3.3. This solution does not make use of any prior knowledge of the target. This implies that there is no knowledge regarding the mass properties of the target; therefore, no prior moments of inertia or centre of mass information is available. There is also no appearance information or known fiducials on the target. Robust features are identified that can be tracked reliably while they are in the field of view. This makes the proposed solution powerful in the sense that it can be used to track the pose of unknown objects. The solution is not limited to space technology and can be used in other robotic problems such as autonomous pick-and-placing [31; 32].

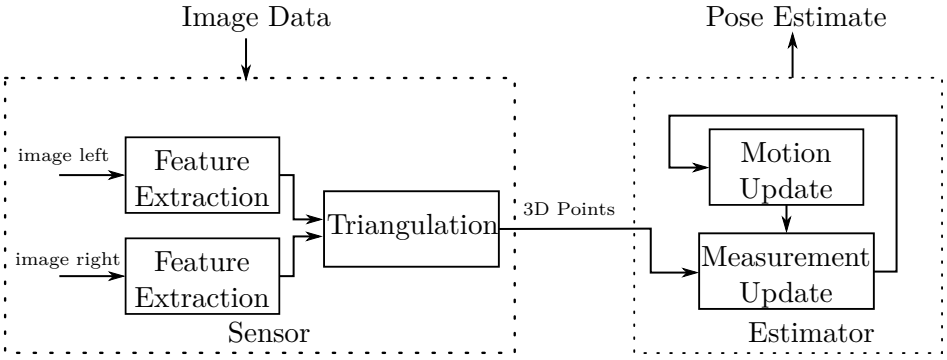


Figure 3.3 – System framework block diagram.

The solution makes use of well known mechanics introduced in Section 3.4 to model the motion of the target. It is assumed that the sensor and target are in the same orbit and that the sensor is in a rendezvous phase with the target, as discussed in Section 2.4. It is further assumed that the target is rigid and there are no external forces or torques present on the target. The total energy of the target will thus remain constant.

3.3 Recursive Estimation

This section describes the elements of recursive estimators along with a few commonly-used estimators in the fields of localisation and tracking. A recursive estimator uses the previous distribution over a set of system states and current sensor data to estimate the current state distribution. Figure 3.4 illustrates the basic workings of a discrete recursive filter.

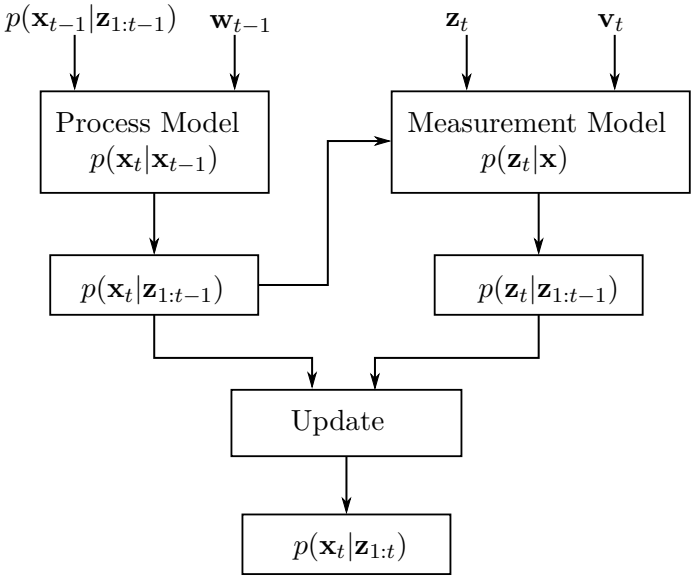


Figure 3.4 – Recursive estimator algorithm flow chart [33].

The state vector is represented by $\mathbf{x}_t$ for the estimation problem in the discrete time domain. The process, or state transition function, is expressed as

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}, \mathbf{w}_{t-1}), \tag{3.3.1}$$

where $\mathbf{f}$ is either a linear or non-linear transition function and $\mathbf{w}_t$ represents the process noise. New observation data, $\mathbf{z}_t$, is available at discrete timesteps and can be related to $\mathbf{x}_t$ by the measurement function,

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t, \mathbf{v}_t). \tag{3.3.2}$$

The measurement uncertainty is represented by $\mathbf{v}_t$ and $\mathbf{h}$ is the observation model, which can also either be linear or non-linear. The goal is to obtain the posterior distribution, $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, over the state vector $\mathbf{x}_t$. This is done by recursively performing the process and measurement updates.

At a time t, the posterior distribution over $\mathbf{x}_{t-1}$ at time $t-1$ is known and the prior distribution at t is calculated as [34],

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}. \tag{3.3.3}$$

The measurement update is used to calculate the new posterior, at time t, given the prior state distribution [34] according to Bayes' rule,

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})}. \tag{3.3.4}$$

The Kalman filter is a popular estimator used in pose estimation problems. It is a special case of the Bayes filter, where Gaussian noise distributions are assumed [34]. The assumption is also made that the initial distribution of the system can be represented by a Gaussian distribution. A control and measurement update is executed at each sampling instant to update the distribution over the states. If the previous state distribution is Gaussian, then the updated current distribution will also be Gaussian and therefore the best estimate is chosen as the mean of the distribution.

Different variants of the Kalman filter exist, of which the extended and unscented Kalman filters are the most popular, each with their own unique characteristics.

- The extended Kalman filter (EKF) overcomes the restrictions of the linear filter by approximating non-linear functions to be linear using a first-order Taylor expansion. The mean position of the state vector is used as the linearisation point around which the tangent of the non-linear function is calculated, allowing the use of standard Kalman filter equations. It is typically more efficient than other non-linear filters which sometimes comes at a cost of reduced accuracy.
- The unscented Kalman filter (UKF) uses stochastic linearisation to deal with non-linear systems. Given a distribution with known mean and covariance, a set of weighted points, known as sigma points, are chosen and transformed using the non-linear function. A new distribution is determined from the transformed sigma points. The process and observation functions do not need to be differentiable and the output is based on values in a larger region, rather than a local approximation.

Kalman filters are well suited for localisation problems since the nature of these systems are normally non-linear. Both the linear and non-linear variants of the Kalman filter are concerned with estimating states using motion and measurement models fused with sensory data. Kalman filters use a system model to perform this data fusion. The following equations are used to model the system [34]:

$$\begin{aligned} \mathbf{x}_t &= A_t\mathbf{x}_{t-1} + B_t\mathbf{u}_{t-1} + \boldsymbol{\epsilon} \\ \mathbf{z}_t &= C_t\mathbf{x}_t + \boldsymbol{\zeta} \end{aligned}, \tag{3.3.5}$$

where

$$\begin{aligned} \boldsymbol{\epsilon} &= \mathcal{N}(\mathbf{0}; R) \\ \boldsymbol{\zeta} &= \mathcal{N}(\mathbf{0}; Q). \end{aligned} \tag{3.3.6}$$

The matrices R and Q are the known covariance matrices of the process and observation noise, respectively and the matrices A, B and C form part of the linear functions. In Algorithm 1, Thrun *et al.* [30] describe the calculation of the updated distribution over $\mathbf{x}_t$ as,

$$p(\mathbf{x}_t|\mathbf{u}_{1:t}, \mathbf{z}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t; \Sigma_t). \tag{3.3.7}$$

Lines 1 to 3 describe the process update. The measurement update is performed in lines 4 to 6 to calculate the posterior distribution, after which the updated mean and covariance is returned in line 7.

Algorithm 1 Kalman Filter

1: Given known $\boldsymbol{\mu}_{t-1}$ and Σ_{t-1}:
2: $\bar{\boldsymbol{\mu}}_t = A\boldsymbol{\mu}_{t-1} + B\mathbf{u}_t$
3: $\bar{\Sigma}_t = A_t\Sigma_{t-1}A_t^T + R_t$
4: $K_t = \bar{\Sigma}_t C_t^T(C_t\bar{\Sigma}_t C_t^T + Q_t)^{-1}$
5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - C_t\bar{\boldsymbol{\mu}}_t)$
6: $\Sigma_t = (I - K_tC_t)\bar{\Sigma}_t$
7: **return** $\boldsymbol{\mu}_t, \Sigma_t$

The state estimation problem in this project makes use of non-linear system models, and a high-dimensional state space. The EKF is well suited for this problem, since it accommodates non-linear process and observation models and is capable of dealing with high-dimensional state spaces. The EKF is often used for the SLAM problem and is well known in the field of robotics and localisation. The EKF is chosen for this project and is discussed in greater depth in Section 5.4.

3.4 Rigid Body Mechanics

3.4.1 Kinematics

The pose of a rigid body in a reference frame consists of the position and attitude of the body. The attitude, or orientation of a body-fixed reference frame with respect to a known reference frame, is usually represented by a rotation matrix, often referred to as a direction cosine matrix (DCM) [35]. A rotation about a single coordinate axis is referred to as a coordinate rotation. A coordinate rotation about the x-, y- and z-axes with angles ϕ, θ and ψ, of a body can respectively be described as,

$$\begin{aligned} R_x(\phi) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \\ R_y(\theta) &= \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \\ R_z(\psi) &= \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{3.4.1}$$

Any rotation in 3D space can be described by three coordinate rotations. The DCM describing the attitude of the target in the camera reference frame (CRF), $\mathbf{A}_{\mathcal{C}}^{\mathcal{B}}$, can be represented by three Euler angles. Each of the angles corresponds to one coordinate rotation. The order of the Euler 1-2-3 rotation, shown in Figure 3.5, is expressed as,

$$\begin{aligned}\mathbf{A}_{\mathcal{C}}^{\mathcal{B}} &= R_x(\phi)R_y(\theta)R_z(\psi) \qquad (3.4.2)\\ &= \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \\ &= \begin{bmatrix} C\theta C\psi & S\theta C\psi & -S\theta \\ S\theta S\phi C\psi - C\phi S\psi & S\theta S\phi S\psi + C\phi C\psi & C\theta S\phi \\ S\phi S\psi + S\theta C\phi C\psi & S\theta C\phi S\psi - S\phi C\psi & C\theta C\phi \end{bmatrix}, \qquad (3.4.3)\end{aligned}$$

where S is the sine function and C the cosine function. The Euler angles are calculated as

$$\begin{aligned}\phi &= \arctan 2\left(\frac{a_{2,3}}{a_{3,3}}\right), \\ \theta &= \arctan 2\left(\frac{-a_{1,3}}{\sqrt{a_{1,1}^2 + a_{1,2}^2}}\right), \text{ and} \qquad (3.4.4)\\ \psi &= \arctan 2\left(\frac{a_{1,2}}{a_{1,1}}\right).\end{aligned}$$

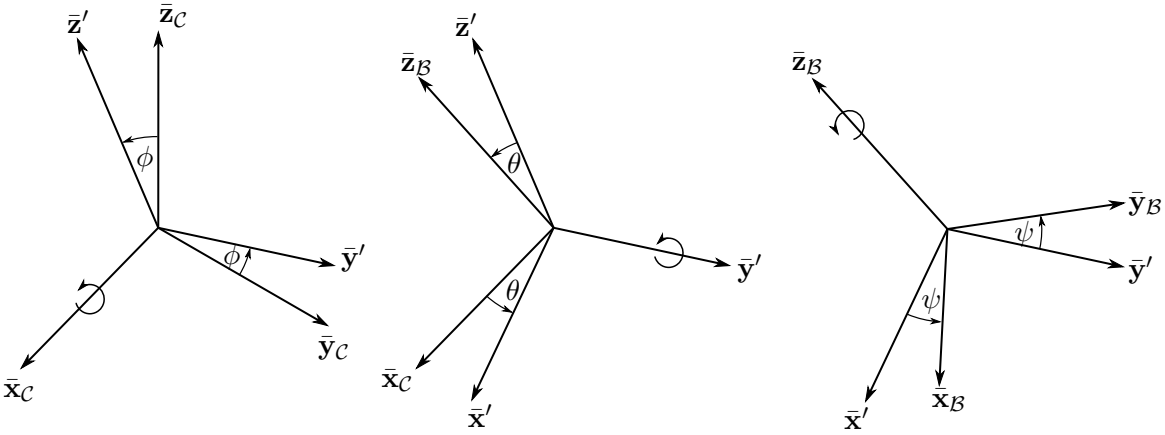


Figure 3.5 – Euler 1-2-3 rotation

Mathematical singularities occur when using Euler angles to represent large rotations. When both $a_{1,1}$ and $a_{1,2}$ in Equation 3.4.2 are zero, the expressions for ψ

and θ are undefined. This is known as *gimbal lock*, where the changes in the first and third Euler angles are indistinguishable when the second angle nears a critical value. Alternatively, the DCM can be described using quaternions, which do not have these singularities. The quaternion rotation in Figure 3.6 is expressed by the Euler axis $\bar{\mathbf{e}} = [e_x, e_y, e_z]^T$ and an angle θ,

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ e_x \sin(\theta/2) \\ e_y \sin(\theta/2) \\ e_z \sin(\theta/2) \end{bmatrix}. \tag{3.4.5}$$

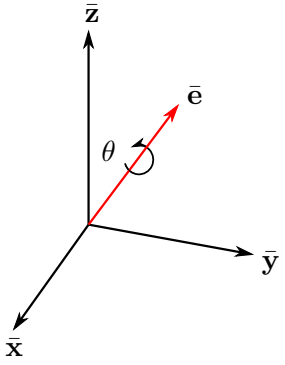


Figure 3.6 – Quaternion rotation

The DCM as a function of a quaternion set is expressed as,

$$\mathbf{A}_{\mathcal{C}}^{\mathcal{B}} = \begin{bmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2(q_2q_3 - q_1q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_4 + q_2q_3) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_1q_2 + q_3q_4) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{bmatrix}. \tag{3.4.6}$$

Using the normalisation constraint, $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$, the DCM is simplified to,

$$\mathbf{A}_{\mathcal{C}}^{\mathcal{B}} = \begin{bmatrix} 1 - 2q_3^2 - 2q_4^2 & 2(q_2q_3 - q_1q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_4 + q_2q_3) & 1 - 2q_2^2 - 2q_4^2 & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_1q_2 + q_3q_4) & 1 - 2q_2^2 - 2q_3^2 \end{bmatrix}. \tag{3.4.7}$$

The body-fixed angular rates of the target in the CRF, $\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}$, is expressed as a function of quaternion rates by,

$$\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}} = \begin{bmatrix} \omega_{bx} \\ \omega_{by} \\ \omega_{bz} \end{bmatrix} = 2 \begin{bmatrix} -q_2 & q_1 & -q_4 & q_3 \\ -q_3 & q_4 & q_1 & -q_2 \\ -q_4 & -q_3 & q_2 & q_1 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix}. \tag{3.4.8}$$

Inversely the quaternion rates as a function of the body rates are,

$$\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \\ \dot{q_4} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\omega_{bx} & -\omega_{by} & -\omega_{bz} \\ \omega_{bx} & 0 & \omega_{bz} & -\omega_{by} \\ \omega_{by} & -\omega_{bz} & 0 & \omega_{bx} \\ \omega_{bz} & \omega_{by} & -\omega_{bx} & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}. \tag{3.4.9}$$

Quaternions will be used throughout this thesis for attitude representations. Quaternions do not have ambiguity regarding the order of rotations and the rotation is around a well-defined axis. The sin and cos elements of the rotation matrix are already encoded in the quaternion form of the DCM. Therefore, only one matrix operation is required for attitude transforms, where Euler angles require three.

3.4.2 Dynamics

Newton-Euler equations are used to describe the rotational dynamics of the target and it is applicable to all rigid inertial bodies [36]. For this project, it is assumed that the angular momentum of the target remains constant and is expressed as,

$$\dot{\mathbf{H}} = \frac{d\mathbf{H}}{dt} = \mathbf{I}\dot{\boldsymbol{\omega}}, \tag{3.4.10}$$

where $\mathbf{H}$ is the angular momentum and the diagonalised moment of inertia tensor is represented by $\mathbf{I}$. When there are no external forces present, the rotational kinematics of a rigid body around its centre of mass and around its principal axis of inertia can be written as,

$$\begin{aligned} I_{xx}\dot{\omega}_x &= \omega_y\omega_z(I_{yy} - I_{zz}), \\ I_{yy}\dot{\omega}_y &= \omega_x\omega_z(I_{zz} - I_{xx}) \quad \text{and} \\ I_{zz}\dot{\omega}_z &= \omega_x\omega_y(I_{xx} - I_{yy}), \end{aligned} \tag{3.4.11}$$

with I_{xx}, I_{yy} and I_{zz} constant and dependent on the shape and mass distributions of the body. The magnitudes of the moments of inertia depend on the mass distribution of the target. It is worthwhile to discuss the stability behaviour of an object due to its mass distribution. It is stated by Marsden and Ratiu [37] that the rotation of a free rigid body is stable around its longest and shortest axis and rotation about the middle axis will not be stable. If the energy in the system is assumed to be constant, then the energy is only distributed over the three principal body rates. Spin about the intermediate axis is not stable and over time the energy is dissipated to the major and minor axes. If the angular velocity is about a non-principal axis, a nutation will be superimposed on the rotation.

Newton's second law of motion describes the linear motion of an object with mass, m and the exact discrete solutions for the displacement d_t and velocity v_t is expressed as,

$$d_t = d_{t-1} + v_t\delta t + \frac{1}{2m}\mathbf{F}(t)\delta t^2 \tag{3.4.12}$$

$$v_t = v_{t-1} + \frac{1}{m}\mathbf{F}(t)\delta t, \tag{3.4.13}$$

where $\mathbf{F}(t)$ is the force working on the object. Since no external torques and forces are assumed and the mass is unknown, the last term in Equations 3.4.12 and 3.4.13 are eliminated. The linear velocity of the target at the current timestep will thus only depend on the previous velocity.

3.5 Conclusion

A formal definition of the project problem and proposed solution was provided in this chapter. The SLAM approach was discussed and how it is used to solve the pose estimation problem of a rotating and translating rigid body. Recursive estimators and Kalman filters were investigated. Mechanics of moving bodies in 3D space were discussed.

The rotational motion of the target implies that a non-linear model will be required to describe the process update of the system. Using a dynamic process model is not possible without information of the relative inertias. Therefore, a kinematic estimator will be implemented, even though it might lead to slower convergence or delays in rate estimates. Using a dynamic estimator with inaccurate inertia information may lead to severe errors in the estimation of all target states.

This project aims to verify the use of an enhanced SLAM solution to solve the autonomous rendezvous problem for space-based applications. Attention is given to the development of the stereo camera sensor in the next chapter and how features on the target surface can be extracted for use in the EKF-SLAM algorithm.

Chapter 4

Image Processing

This chapter focuses on sensor modelling and measurement extraction using a stereo vision sensor. A feature-based method is used, where the position of reference points on the target are tracked over time. It is necessary that these distinct image points are extracted consistently so that correspondence can be established. The monocular pinhole camera model is introduced to describe how 3D world coordinates are related to the CRF. Stereo geometry and depth calculation is discussed and a sensor model is derived. The feature class and methods for how features are detected and matched using image processing techniques are introduced.

4.1 Pinhole Camera Model

It is necessary to model the transformation from 3D world points to 2D image plane coordinates. The pinhole camera is an ideal model that adequately represents this perspective transformation. An ideal camera assumes all light enters through a pinhole or single point known as the optical centre. The light produces a vertically flipped image that is projected onto an image plane behind the optical centre, at a focal distance, f. The image plane can also be placed in front of the optical centre which results in an upright projection. The image created in the pinhole camera is perfectly focused, the aperture is infinitely small and no geometric lens distortion exists.

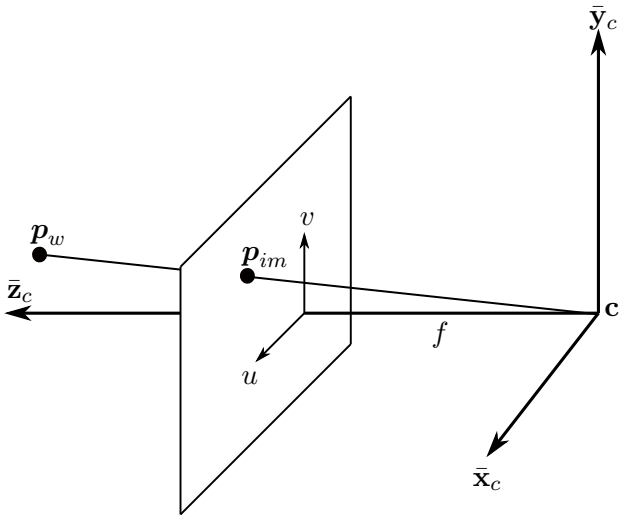


Figure 4.1 – Pinhole camera model

Figure 4.1 shows a pinhole camera model with the principal axis in line with the $\bar{\mathbf{z}}_c$-axis and the centre of projection, $\mathbf{c}$ at the origin of the coordinate system. A point, $\boldsymbol{p}_w = [x_w, y_w, z_w]^T$, in 3D space is projected onto the image plane at $\boldsymbol{p}_{im} = [u, v]^T$. Using similar triangles, $\boldsymbol{p}_{im}$ can be described as

$$\boldsymbol{p}_{im} = \begin{bmatrix} f\dfrac{x_w}{z_w} & f\dfrac{y_w}{z_w} \end{bmatrix}^T . \tag{4.1.1}$$

Using homogeneous vectors, the perspective transformation from $\boldsymbol{p}_w$ to $\boldsymbol{p}_{im}$ is defined as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = [K \quad \mathbf{0}] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} , \tag{4.1.2}$$

where

$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{4.1.3}$$

The 3×3 projection matrix K, assumes that the origin of the image coordinates is at the principal point where the $\bar{\mathbf{z}}_c$-axis intersects the image plane. It is not always the case in practical systems and therefore translation elements, p_x and p_y, are added to K. Scaling changes between the world and image coordinate systems, m_x and m_y, are also accounted for, expanding K to

$$K = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.1.4}$$

The focal length of the camera in terms of pixel coordinates is represented by $\alpha_x = fm_x$ and $\alpha_y = fm_y$ and the principal point in pixels is represented by $x_0 = m_x p_x$ and $y_0 = m_y p_y$ in Equation 4.1.4. A skewing parameter s, is added to adjust for shear distortion of the projection on the image plane.

Parameters have now been added to K to account for translation, scaling and shear of the image plane. K is also known as the intrinsic matrix and describes parameters that are unique to the camera system. This is developed under the assumption that the CRF is aligned with the world coordinate system. Normally, it is desired to describe the image in terms of the coordinate system that is chosen for the scene, which is not necessarily the camera's coordinates. Therefore extrinsic parameters are required to describe the relative rotation and translation between the reference frames.

The CRF is translated with $\mathbf{t} = [t_x, t_y, t_z]^T$ to the origin of the scene coordinate system. A rotation, R, can then be applied to coincide the principal axis with the $\bar{\mathbf{z}}_c$-axis. The complete transformation is represented as,

$$\boldsymbol{p}_{im} = P\boldsymbol{p}_w, \tag{4.1.5}$$

where

$$P = KR[I \quad \mathbf{t}]. \tag{4.1.6}$$

Camera calibration techniques allow the estimation of the camera matrices. The methods proposed by Zhang [38] are often used to estimate these camera parameters. Two calibrated pinhole cameras are used in the section that follows to model the stereo-camera sensor used in this project.

4.2 Stereo Geometry

Using two cameras to observe the same scene enables the calculation of depth from the images. This section develops the stereo geometry required to describe 3D points in the scene, given 2D image plane coordinates of the points.

Consider two calibrated pinhole cameras with calibration matrices K_L, R_L, K_L and R_L with centres $\mathbf{c}_L$ and $\mathbf{c}_R$, where L and R denotes the left and right camera, respectively. Given a point $\boldsymbol{p}_L$ in the left image plane and a point $\boldsymbol{p}_R$ in the right, the corresponding 3D point $\boldsymbol{p}_w = [x_w, y_w, z_w]^T$, that is observed by both cameras, can be calculated. Using Equation 4.1.5 the points $\boldsymbol{p}_L$ and $\boldsymbol{p}_R$ are described as,

$$\boldsymbol{p}_L = K_L R_L [I \quad \mathbf{c}_L] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{p}_R = K_R R_R [I \quad \mathbf{c}_R] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \tag{4.2.1}$$

where $\boldsymbol{p}_L = [x_L, y_L, 1]^T$ and $\boldsymbol{p}_R = [x_R, y_R, 1]^T$. Figure 4.2 shows the stereo camera pair with the point $\boldsymbol{p}_w$ projected onto the image planes at $\boldsymbol{p}_L$ and $\boldsymbol{p}_R$. The baseline is a distance b between the optical centres. The plane between $\mathbf{c}_L$, $\mathbf{c}_R$ and $\boldsymbol{p}_w$ is referred to as the epipolar plane. Epipolar lines are the intersection lines between the epipolar plane and the image plane.

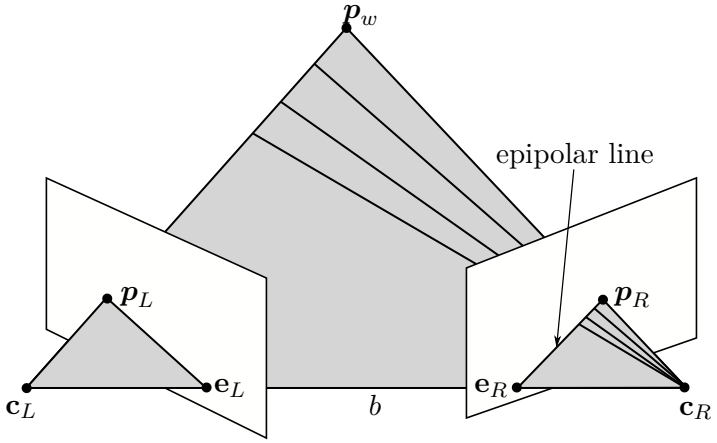


Figure 4.2 – Epipolar geometry between two cameras.

Using the geometry of the epipolar planes and epipolar lines, the 3×3 fundamental matrix F, can be derived that relates any point $\boldsymbol{p}_L$ to its corresponding point $\boldsymbol{p}_R$ according to

$$\boldsymbol{p}_R^T F \boldsymbol{p}_L = 0. \tag{4.2.2}$$

The process of searching for point matches in stereo frames can be further simplified using image rectification. The goal of rectification is to transform the images so that they share the same intrinsic properties and that the only difference between the images will be the baseline offset. The transformation will result in the epipolar lines becoming collinear and parallel to the $\bar{\mathbf{x}}_{rect}$-axis. Given the camera matrices,

$$P_L = K_L R_L [I \quad \mathbf{c}_L] \qquad \text{and} \qquad P_R = K_R R_R [I \quad \mathbf{c}_R], \tag{4.2.3}$$

the common KR is desired so that the transformed camera matrices become,

$$P_L^{'} = KR[I \quad \mathbf{c}_L] \qquad \text{and} \qquad P_R^{'} = KR[I \quad \mathbf{c}_R]. \tag{4.2.4}$$

This is done by first arbitrarily choosing K as the average between the intrinsic matrices of the cameras,

$$K = \frac{1}{2}(K_L + K_R). \tag{4.2.5}$$

Choosing the average limits distortion brought on by the transformation. Brink [29] has shown that each column of the rotation matrix, R, can be calculated separately. The second row corresponds to the rectified $\bar{\mathbf{y}}_{rect}$-axis and must be parallel to the baseline,

$$\mathbf{r}_2 = \frac{\mathbf{c}_R - \mathbf{c}_L}{||\mathbf{c}_R - \mathbf{c}_L||}. \tag{4.2.6}$$

The transformed $\bar{\mathbf{z}}_{rect}$-axis must be orthogonal to the principal axis of the left camera,

$$\mathbf{r}_3 = \frac{\mathbf{u} \times \mathbf{r}_2}{||\mathbf{u} \times \mathbf{r}_2||}, \tag{4.2.7}$$

where $\mathbf{u}$ is the unit vector on the left principal axis. The unit vector corresponding to $\bar{\mathbf{x}}_{rect}$ must be orthogonal to $\mathbf{r}_2$ and $\mathbf{r}_3$. Column $\mathbf{r}_1$ is calculated as,

$$\mathbf{r}_1 = \mathbf{r}_2 \times \mathbf{r}_3. \tag{4.2.8}$$

The images are transformed by projecting the points to the old image plane and then reprojecting them to the new image plane points, $\boldsymbol{p}_L^{'}$ and $\boldsymbol{p}_R^{'}$, using the common camera matrices, K and R,

$$\boldsymbol{p}_L^{'} = KR[K_L R_L]^{-1}\boldsymbol{p}_L \qquad \text{and} \qquad \boldsymbol{p}_R^{'} = KR[K_R R_R]^{-1}\boldsymbol{p}_R, \tag{4.2.9}$$

where

$$R = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix}. \tag{4.2.10}$$

Stereo rectification ensures that all image correspondences will have the same vertical coordinates. Rectification simplifies triangulation and depth calculation of points.

Using knowledge from the pinhole camera model, intrinsic and extrinsic camera parameters and stereo geometry, it is possible to establish the position of a 3D point. Consider the rectified stereo camera configuration in Figure 4.3. The point $\boldsymbol{p}_w$ intersects the left and right image planes at $\boldsymbol{p}_L = [x_L, y_L]^T$ and $\boldsymbol{p}_R = [x_R, y_R]^T$ respectively. The depth z_w is calculated using triangulation. Let the disparity be defined as,

$$d = x_L - x_R, \tag{4.2.11}$$

where x_L and x_R are the horizontal positions of the projection of $\boldsymbol{p}_w$ in the left and right images. Choosing the origin of the stereo camera coordinate system to coincide with the left camera, x_L and x_R can be described as,

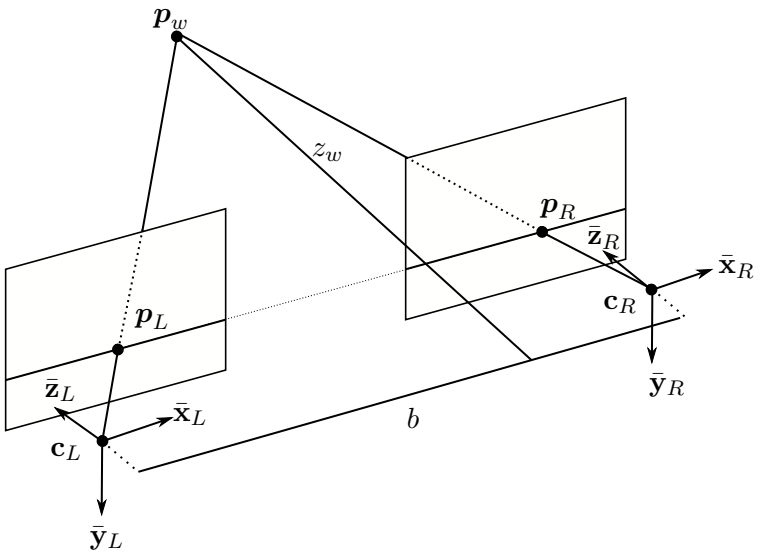


Figure 4.3 – Rectified stereo geometry

$$x_L = f\frac{x_w}{z_w} \qquad \text{and} \qquad x_R = f\frac{x_w - b}{z_w}. \tag{4.2.12}$$

By substituting Equation 4.2.12 into 4.2.11, z_w is calculated,

$$z_w = f\frac{b}{d}. \tag{4.2.13}$$

The horizontal and vertical position of the 3D point is calculated using similar triangles,

$$x_w = \frac{x_L z_w}{f} \tag{4.2.14}$$

$$y_w = \frac{y_L z_w}{f}. \tag{4.2.15}$$

A stereo-camera model has been derived and is used to determine the 3D position of any world point, given two point correspondences of the world point in the left hand and right hand images.

4.3 Measurement Extraction

This section introduces the procedure followed to extract measurements from stereo images. These measurements are used in the estimation algorithm to track the states of the target. Monocular vision is used to follow a reference set of features. Only the left camera stream is used for feature identification and feature extraction. The right camera is regarded as an auxiliary sensor that is used for depth calculation

and to calculate the 3D world coordinates of the reference features according to Equations 4.2.13, 4.2.14 and 4.2.15.

Figure 4.4 shows the basic scheme of the measurement extraction framework. The process of extracting measurements from stereo-camera images works as follows. Firstly, the system is initialised and a reference feature set, denoted by F_R, is created. Initialisation of F_R is discussed in Section 4.3.1. Secondly, for each timestep after the initialisation, features in the left hand frames are detected and compared to the reference set. The reference set is updated using the newly detected features. The development of the image plane feature extractor is discussed in Section 4.3.2.

A stereo matching algorithm is developed in Section 4.3.7 that runs concurrently to the image plane feature extractor. Features are extracted from the right hand camera frames and matched to the reference feature set, F_R. A measurement set, $\mathbf{z}_t$, at timestep t, is formed if valid point correspondences are found between the features in F_R and the right hand frame features.

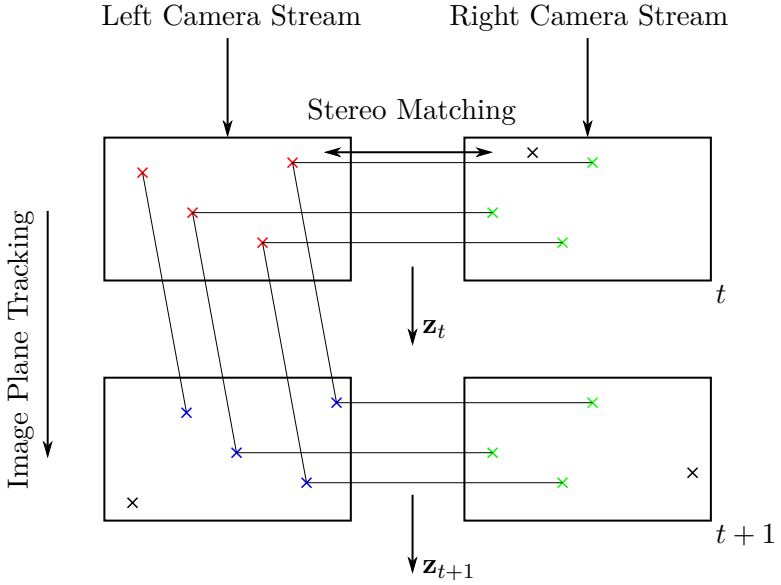


Figure 4.4 – Measurement extraction framework. Features are detected in the left frame and matched in the right frame. Calculation of the real world points in 3D coordinates are calculated using these matches. The positions of features in $F_{R,t}$ are shown with red crosses. The updated feature set, $F_{R,t+1}$, is displayed with blue crosses. The positions of stereo matches are indicated in green.

4.3.1 Reference Feature Set Initialisation

In order to consistently and reliably extract features over time, a data structure that stores all the information of the features is introduced. A feature consists of three parts: a keypoint that stores the position of the feature in the image plane, a descriptor that provides pixel based appearance information about the area around the keypoint, and a label to keep track of correspondence. The detection of scale invariant feature transform (SIFT) keypoints and descriptors is discussed in more

depth in Section 4.3.5. Figure 4.5 shows the data structure of the reference feature set, F_R.

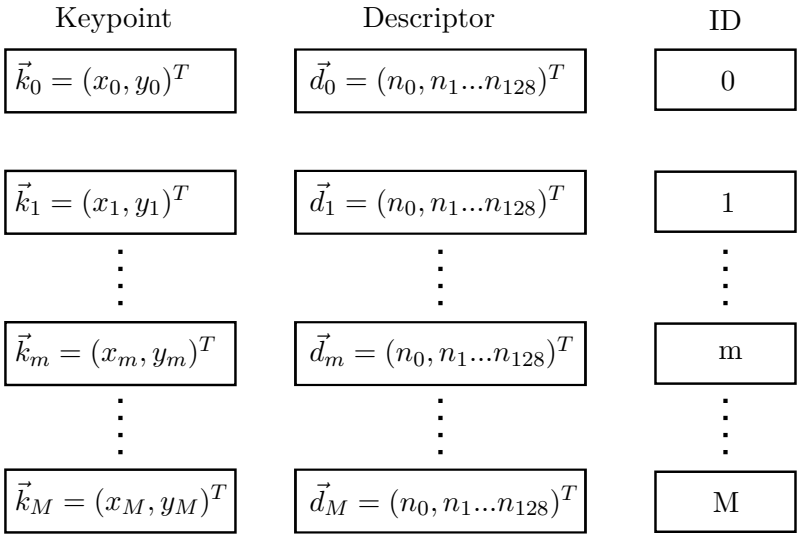


Figure 4.5 – The reference feature set, F_R

When the feature tracker is initialised, all the identified features in the left camera image are copied into the reference data structure. Labels are assigned from 0 to M, where M is the number of identified features.

4.3.2 Image Plane Feature Extraction

A diagram illustrating the algorithm for extracting features in the left hand image plane is shown in Figure 4.6. When a new frame, $I_{L,t+1}$, is available, optical flow is calculated using the previous frames $I_{L,t-1}$ and $I_{L,t}$, to predict the regions in which the features in $F_{R,t}$ will be located in $I_{L,t+1}$. The optical flow architecture is discussed in Section 4.3.3.

A region of interest (ROI) is calculated to limit the feature detection search region. An in-depth discussion of the ROI calculation is provided in Section 4.3.4. Feature detection is performed on $I_{L,t+1}$ and each detected feature is compared to the ones in the reference set, $F_{R,t}$, to find a possible match for it. Feature detection is described in Section 4.3.5. If a match is found, the keypoint and descriptor of that feature is updated with the new feature's information in $F_{R,t+1}$. Section 4.3.6 discusses how a brute-force matching algorithm is used to compare features.

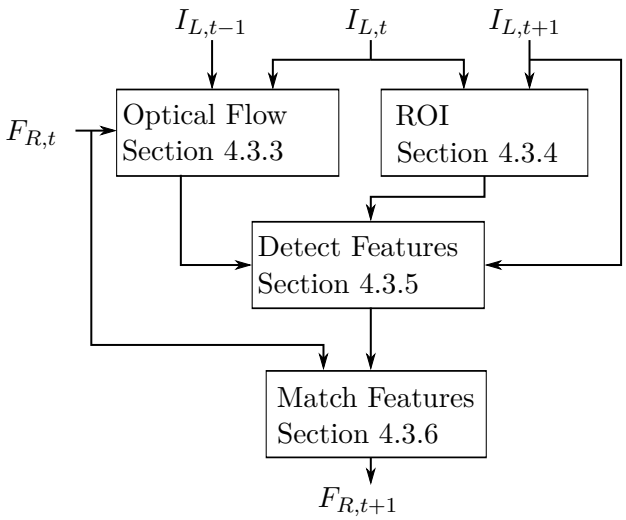


Figure 4.6 – Image plane feature extraction algorithm

The subsections that follow will discuss each of the blocks from Figure 4.6 separately.

4.3.3 Optical Flow

Optical flow is the patterns formed by the motion of features in a sequence of images in a scene. Optical flow is often used for applications like structure from motion (SFM) and video compression and stabilisation [39]. Using discrete displacements of the keypoints, it is possible to predict the possible positions of the keypoints in subsequent frames, narrowing the search area of matches in the following frame. It works on the assumption that neighbouring pixels all have the same motion and that pixel intensities are translated between consecutive frames [40] according to

$$I(\mathbf{x}, t) = I(\mathbf{x} + \mathbf{v}, t + 1), \tag{4.3.1}$$

where the pixel intensity $I(\mathbf{x}, t)$ is a function of the position $\mathbf{x} = [x, y]^T$, time t and the pixel velocity $\mathbf{v}$.

A prediction of where the keypoints will be located in the future image is made using a first-order Taylor series expansion about $I(x, y, t)$,

$$I(x + \delta_x, y + \delta_y, t + \delta_t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta_x + \frac{\partial I}{\partial y}\delta_y + \frac{\partial I}{\partial t}\delta_t. \tag{4.3.2}$$

Using Equations 4.3.1 and 4.3.2, it can be shown that

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0, \tag{4.3.3}$$

where $v_x = \frac{\delta_x}{\delta_t}$ and $v_y = \frac{\delta_y}{\delta_t}$. The Lucas-Kanade method [41] is a popular solution to the optical flow problem stated in Equation 4.3.3. The image gradients, $\frac{\partial I}{\partial x}$ and

$\frac{\partial I}{\partial y}$ and time gradient $\frac{\partial I}{\partial t}$ can be calculated, but the velocity vector $\mathbf{v} = [v_x, v_y]^T$ is unknown and cannot be solved directly. Because it is assumed that neighbouring pixels have similar velocities, they will have similar motion. The Lucas-Kanade method uses a 3×3 pixel patch around $\mathbf{x}$, the point of interest, to compute this motion. The image and time gradients for these 9 points are calculated and a least-squares fit is used to solve for $\mathbf{u}$ as follows,

$$\mathbf{u} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum \frac{\delta_I}{\delta_x}^2 & \sum \frac{\delta_I}{\delta_x}\frac{\delta_I}{\delta_y} \\ \sum \frac{\delta_I}{\delta_x}\frac{\delta_I}{\delta_y} & \sum \frac{\delta_I}{\delta_y}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum \frac{\delta_I}{\delta_x}\frac{\delta_I}{\delta_t} \\ -\sum \frac{\delta_I}{\delta_y}\frac{\delta_I}{\delta_t} \end{bmatrix}. \tag{4.3.4}$$

The pixel velocity vector is integrated to predict where a feature will be located in the next frame, which reduces the search area and makes the feature extraction algorithm more robust and efficient.

4.3.4 Determining a Region of Interest

Creating a region of interest (ROI) limits the area in which features are extracted. Target segmentation using a ROI has two advantages, it firstly limits the search region which decreases computation time and secondly, the ROI will constrain features to the target only, limiting possible mismatches and outliers. Several operations can be performed on the image in order to isolate the object of interest. This section briefly describes the process of extracting such a ROI of the target.

The first preprocessing step taken to isolate the object is background subtraction. If an image of the scene without any objects is available, it is subtracted from the current image. The outcome will be an image that only contains the object. In this project, such an image is not available, since the environment is unknown and ever-changing. Therefore, the image in question is used along with the previous image to perform background subtraction by computing the absolute difference, B, between two images, I_1 and I_2,

$$B(x, y) = |I_1(x, y) - I_2(x, y)|. \tag{4.3.5}$$

It is assumed that there exists relative pixel movements between frames, otherwise the background subtraction operation will result in a solid black image, corresponding to a grayscale image with pixels values of zero. Background subtraction isolates the object, but some areas of the image may still need to be removed. The image is converted from grayscale to binary using a thresholding operation. All pixels with an intensity $I_{x,y} < T$, where T is a predetermined threshold value, are replaced by black ones. White pixels are placed at locations where the intensity exceeds the threshold. Figure 4.7 shows the result after background subtraction and binary thresholding have been applied.

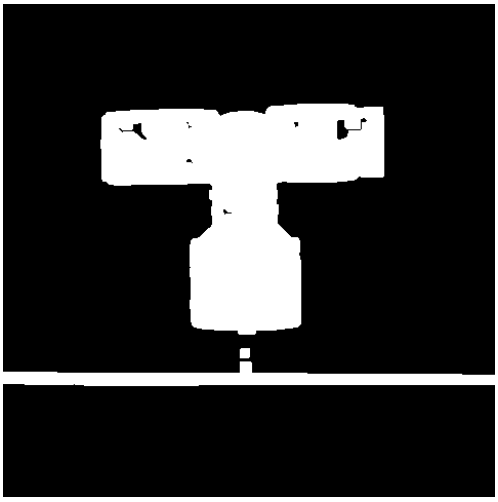


Figure 4.7 – Result after background subtraction and binary thresholding

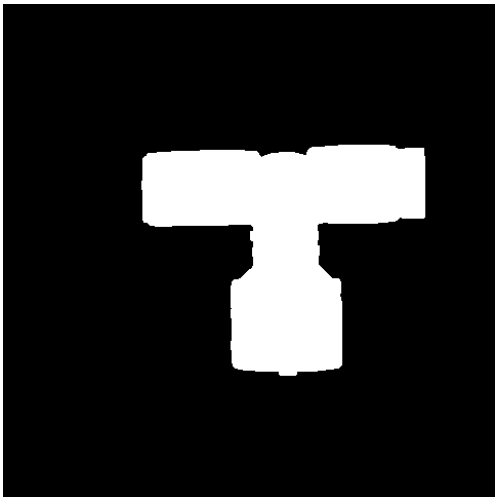


Figure 4.8 – Region of interest (ROI) created from largest set of contours

The final step is to identify contours in the image and assume that the target will be in the contour set with the largest area. The mask shown in Figure 4.8 shows the final ROI of the frame. Feature detection is done only in this region.

4.3.5 Feature Detection

Detecting and describing distinct points of interest in images have long been researched in the field of robotics. It is important that feature detectors are robust against noise, scale and illumination changes. Some of the most noteworthy feature detectors are speeded up robust features (SURF) [42], SIFT [43] and binary robust invariant scalable keypoints (BRISK) [44]. SIFT was chosen for the purpose of this research since SIFT features are accurate, robust to scale and rotation changes and the SIFT method is capable of extracting large quantities of features, even in small areas.

Each SIFT feature consists of a keypoint and a descriptor. Keypoints are located in the image at local minima and maxima regions over different scale spaces. Descriptors are used to identify the keypoint in the future. The descriptor is a vector representing a histogram of intensity gradients of the pixels surrounding the keypoint.

The scale space is used to intentionally remove unnecessary details in the image by blurring the image. The image is progressively blurred by increasing the standard deviation σ, in Equation 4.3.6. The only way to do this without adding false details or noise is by using Gaussian blur. The image I, is convolved with the Gaussian operator according to,

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{4.3.6}$$

where L is the blurred image and σ is the scaling parameter, indicating the amount of blur to be added. G is the Gaussian blur operator which is described as,

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{4.3.7}$$

The image is progressively convolved using the Gaussian blur operator and σ is increased with every convolution to create *octaves*. The last image in the octave is halved in size and a new octave is created by once again progressively blurring the smaller image. Each layer of octaves is referred as a *scale*. Once the scale space is created, scale-space extremum regions can be located using a difference of Gaussian (DoG) operation. The DoG is computed by convolving the image with the difference of two nearby scales,

$$D(x, y, \sigma) = G(x, y, k\sigma) - G(x, y, \sigma) * I(x, y), \tag{4.3.8}$$

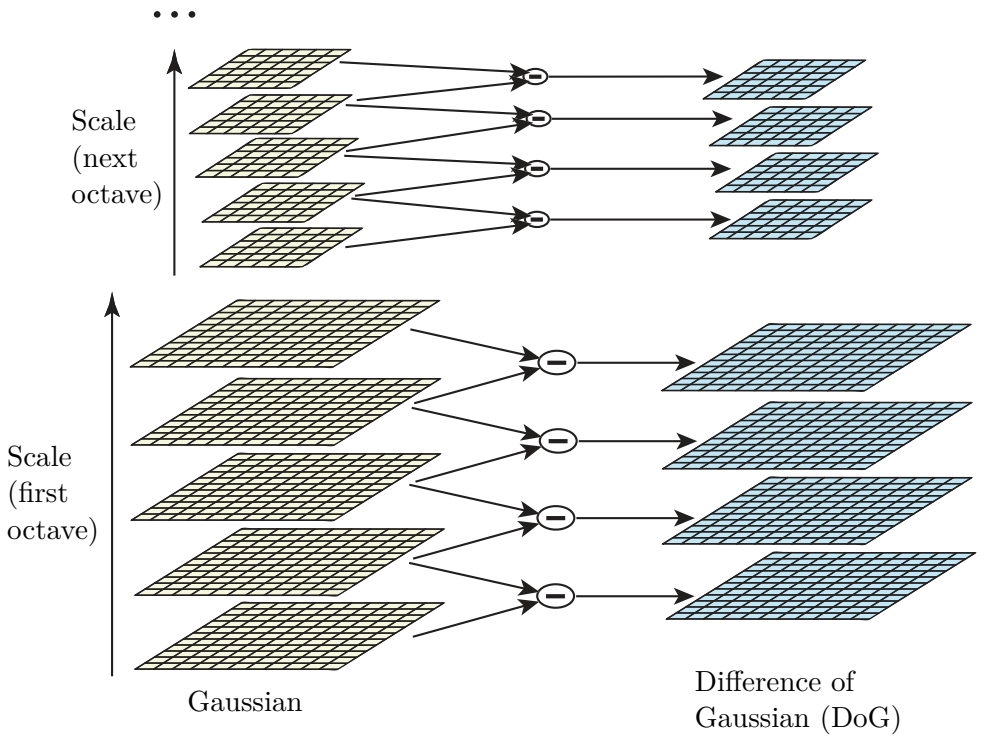


Figure 4.9 – An image scale space that is created by repeatedly convolving the image with Gaussians. The DoG space is created by subtracting adjacent Gaussian images. Image reproduced with permission from Lowe [43].

with a scaling factor k. Figure 4.9 illustrates the calculation of DoGs in the scale space of an image. The DoG is computationally efficient since it relies on simple image subtraction operations. The local extrema are located by comparing neighbouring pixels in the resultant images across scale spaces. Each pixel is compared to all its neighbours, as well as the neighbours in the scales above and below it. A

pixel is regarded as a keypoint if it has the highest or the lowest intensity in its neighbourhood. These keypoints are only approximate, since the maxima or minima are rarely located exactly at the pixel position. Sub-pixel accuracy is calculated using a Taylor-series expansions of the DoG operation, $D(x, y, \sigma)$, around the pixel positions,

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}. \tag{4.3.9}$$

According to Lowe [43], the 2D location of a local extrema point, $\hat{\mathbf{x}}$, can be calculated by taking the derivative of the function in Equation 4.3.9 and setting it to zero. The location of $\hat{\mathbf{x}}$ can then be expressed as

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}}. \tag{4.3.10}$$

Using the DoG operation to find keypoints will often yield points along edges or in low contrast areas. These need to be removed since they are not very robust. Ideally, keypoints are located at corners. Corners are identified by calculating two perpendicular gradients at the keypoint location. If both gradients are large it is considered to be a corner. Edges will have a large gradient perpendicular to the edge and a small one along the edge.

The gradients are assessed by computing the Hessian matrix H at the location and scale of a keypoint,

$$H(x, y, \sigma) = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}. \tag{4.3.11}$$

The eigenvalues of H are proportional to the principal gradients. If the ratio between the eigenvalues are below a predetermined threshold, the keypoint is considered to be at a corner instead of an edge. Therefore it is not necessary to compute the magnitudes of the eigenvalues directly, but rather the ratios between them.

Keypoints need to be invariant to rotation changes. Orientations of the keypoints are determined at their corresponding scales, thus also making them scale invariant. The gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, are computed over the Gaussian smoothed image, L as,

$$m(x, y) = \sqrt{(L(x+1, y, \sigma) - L(x-1, y, \sigma))^2 + (L(x, y+1, \sigma) - L(x, y-1, \sigma))^2} \tag{4.3.12}$$

and

$$\theta(x, y) = \tan^{-1}\left(\frac{L(x, y+1, \sigma) - L(x, y-1, \sigma)}{L(x+1, y, \sigma) - L(x-1, y, \sigma)}\right). \tag{4.3.13}$$

The computed orientations are used to form a histogram. Each histogram bin covers 10 degrees and there are 36 bins. Every sample added to the histogram is weighted

according to its gradient magnitude. Peaks in the histogram indicate the dominant direction of the local gradients. If a second peak exists within 80% of the highest peak, an additional keypoint is added at that location, with the only difference being the orientation. Figure 4.10 shows SIFT keypoints identified on a target.

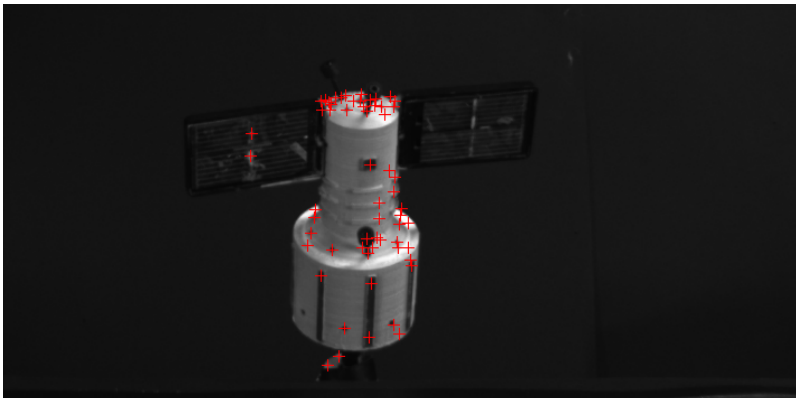


Figure 4.10 – Detected SIFT features

Similarly to computing keypoint orientations, the neighbouring pixel orientations and gradient magnitudes are used to compute descriptors. Figure 4.11 illustrates the computation of a feature descriptor.

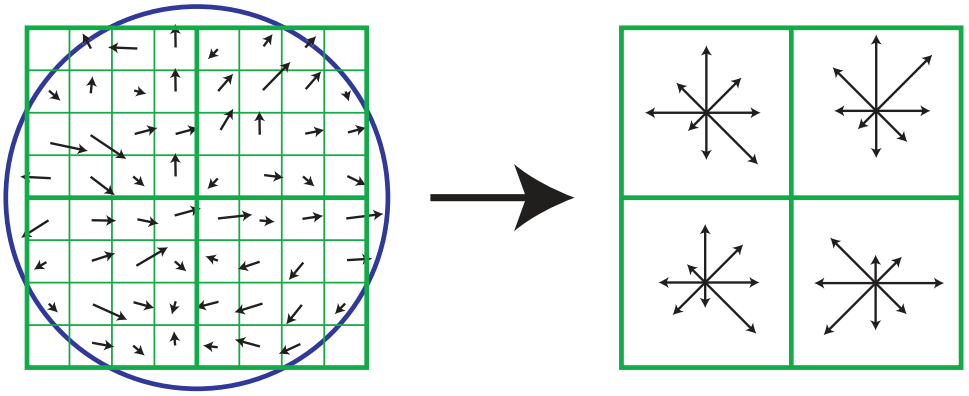


Figure 4.11 – Keypoint descriptor created from histogram binning of image gradients. The circle indicates the Gaussian weighting window. For simplification, this figure uses an 8×8 sampling window with a 2×2 descriptor array. The actual descriptors used in this project are 4×4 descriptor arrays that are computed from a 16×16 sampling window. Image reproduced with permission from Lowe [43].

A 16×16 window is chosen around the keypoint. This window is divided into sixteen 4×4 windows. The gradient magnitudes and orientations within each 4×4 window are placed into an eight-bin histogram, with each bin representing 45 degrees. The orientation of the gradients are rotated by the keypoint orientation, making the

descriptor rotation invariant. The 4×4 regions, each with 8 vectors, result in a $4 \times 4 \times 8 = 128$ dimensional descriptor vector. The influence of gradients on the descriptor is dependent on the distance from the keypoint. A Gaussian weighting function is applied over the histograms. Gradients closer to the keypoint will have a bigger weight than those that are on the edges of the 16×16 window. The descriptor is normalised and to eliminate illumination dependence, any values greater than the illumination threshold are forced to the threshold and the vector is normalised again.

4.3.6 Feature Matching

Feature descriptors are used to match features from one timestep to another in the left hand images for feature extraction and also for stereo matching between left and right camera images. A brute-force matching algorithm is used to calculate the Euclidean distance between every descriptor in the reference feature set to every possible match in the update feature set. The descriptor pair with the smallest distance is chosen as a match. A distance m, between a descriptor vector $\mathbf{u}$ in the reference set and descriptor vector $\mathbf{v}$ in the update set is calculated as

$$m(\mathbf{u}, \mathbf{v}) = \left(\sum_i (u_i - v_i)^2 \right)^{1/2}. \tag{4.3.14}$$

The distance is used to determine the quality of the match. The accuracy and reliability of the matches are improved by performing matches in both directions [43]. If a descriptor $\mathbf{u}$ in the reference set has a match $\mathbf{v}$ in the update set, then $\mathbf{u}$ must also be a match for $\mathbf{v}$ when searching with $\mathbf{v}$ as a reference. A minimum distance threshold is used to remove false matches in the case that a valid match is not available. The minimum distance value is determined empirically.

As mentioned in Section 4.3.7, epipolar constraints and positive disparity checks are implemented to reduce the number of false matches. A bijectivity check is also done to ensure that every feature has only one unique match. The check ensures that the second best match for a feature must be significantly worse than the best match, otherwise both matches are discarded. This eliminates false matches when there are repetitive or symmetric features on the target.

4.3.7 Stereo Matching Algorithm

A stereo matching algorithm is used to find corresponding points between the reference feature set from the left hand frames and features that are found in the right hand frames. The main objective of stereo matching is to calculate the disparity for each pair of corresponding points. The values of the focal length, f, and the baseline b, are already known from calibration. They are used with the disparity to compute the 3D positions of the points. Figure 4.12 shows a single feature from F_R in the left hand frame and a set of possible matches in the right hand frame at a single timestep.

It can be seen from Figure 4.12 that the possible matches in the right hand frame are restricted to a small region. This region is formed using the ROI calculation

Figure 4.12 – A feature in the left hand frame and possible matches in the right hand frame.

from Section 4.3.4. A ROI of the object in the right hand frame is calculated using the current and previous right hand camera images.

Stereo rules are used to further limit the search region. A feature is considered a possible match if it adheres to the two rules. The first is the disparity rule: given two images from a rectified stereo-camera pair, a match can only be valid if the disparity calculation yields a positive number. The second is the rectification rule: Corresponding image plane points from the left hand and right hand frames must lie on the same horizontal line. A threshold is used to account for minor calibration errors when calculating the positions of the points on the horizontal lines. The threshold value is determined empirically.

Algorithm 2 outlines the procedure used to perform stereo matching. The output of the algorithm is a set of measurements, $\mathbf{z}_t$, at every timestep. The measurement set is denoted as

$$\mathbf{z}_t = \begin{bmatrix} \mathbf{z}_{t,1}^T \\ \vdots \\ \mathbf{z}_{t,K}^T \end{bmatrix}, \tag{4.3.15}$$

where $\mathbf{z}_{t,k} = [x_{t,k}, y_{t,k}, z_{t,k}]^T$, and represents the 3D position of the measurement in world coordinates. The most important steps are discussed for clarification:

- **Line 2:** A set of features S_t, is created. The set contains all the SIFT features that are extracted from the right hand frame, I_R, at timestep t. The set is comprised of the keypoints and descriptors for each feature.
- **Lines 3-4:** Retrieve the 2D coordinates, x_F and y_F, of every keypoint in the reference feature set.
- **Lines 5-7:** Retrieve the 2D coordinates x_S and y_S, of the keypoint at position j in the feature set of possible matches.
- **Lines 8-9:** Test if the feature, $S_t[j]$, complies to the two stereo tests. The disparity and rectification rules are tested here. If the feature passes both

tests, it is appended to a list of possible matches for the corresponding feature $F_{R,t}[k]$.

- **Lines 12-14:** The brute-force descriptor matcher, discussed in Section 4.3.6, is used to identify the best match for the feature $F_{R,t}[k]$ once all of the possible matches for it have been identified. If a valid match is found, the corresponding points are used to compute the 3D position of the feature. The 3D point is appended to the measurement set, $\mathbf{z}_t$.

Algorithm 2 Stereo matching algorithm

```
1:  function MATCHFEATURES(F_{R,t}, I_{L,t}, I_{R,t})
2:      S_t ←DetectAndCompute(I_{R,t});
3:      for k ← 1 to length(F_{R,t}) do
4:          x_F, y_F ← F_{R,t}[k]   ▷ Get the coordinates of the feature in the left frame
5:          N ← length(S_t)
6:          for j ← 1 to N do
7:              x_S, y_S ← S_t[j] ▷ Get the coordinates of the feature in the right frame
8:              if x_F > x_S and |y_F − y_S| <thresh then
9:                  Append S_t[j] to list C_k, of possible matches for feature F_{R,t}[k]
10:             end if
11:         end for
12:         M ←Match(F_{R,t}[k], C_k)                          ▷ Select best match
13:         P_{t,k} ←Compute3D(F_{R,t}[k], M)
14:         Append P_{t,k} to z_t
15:     end for
16:     return z_t                  ▷ Measurement set with 3D positions of features
17: end function
```

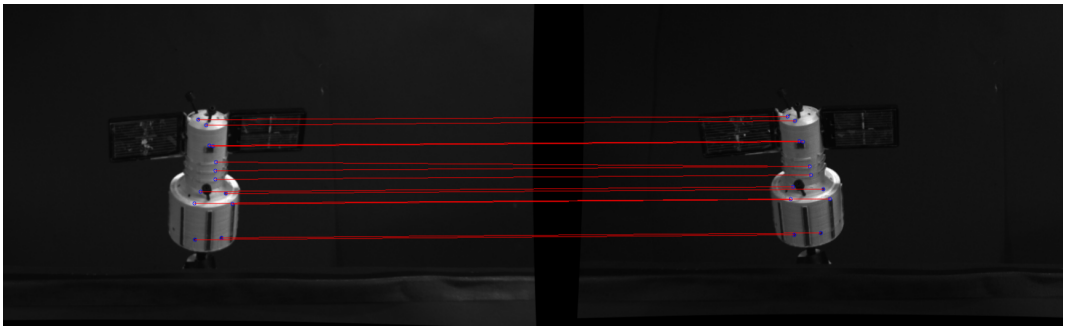


Figure 4.13 – Stereo matching between features in the left and right hand images.

Figure 4.13 shows matched features between the left and right image frames. The ultimate goal of the image plane feature extractor and stereo tracking algorithm is to consistently extract these matches to form measurements for the pose estimation algorithm.

4.4 Conclusion

This chapter presented geometric models for monocular- and stereo-vision systems and how they are used to simplify several image processing problems. The algorithms used in this project to extract sparse features with 3D information were also discussed.

The image plane feature extraction system is able to identify and follow the movement of unique features in the left hand image plane. These features are used to keep track of the feature correspondence. This provides the estimator with a unique label for each tracked point on the target. The stereo matching algorithm allows the calculation of the 3D Cartesian feature positions which are used as measurements in the estimation algorithm.

The feature extraction algorithm relies on the assumption that the target is moving relative to the cameras. Features are detected and tracked only in the ROI. Segmenting the target from the rest of the image is dependent on the relative movement of pixel regions in the background subtraction operation. If there is no relative movement, then no ROI can be formed. Measurements are generated in a simulation environment in the next chapter. This allows for the independent testing of the estimation system.

Chapter 5

State Estimation

5.1 Introduction

This chapter will briefly revisit the Kalman filter equations described in Section 3.3 and approximations are included to account for non-linear systems. The full state vector that describes the target's pose is introduced along with the motion and measurement models to design a tracking algorithm. The functionality of the tracker is tested using a simulation environment and simulation results are presented.

5.2 The Extended Kalman Filter

The Kalman filter was modelled previously with linear motion and measurement models. This was expressed in Section 3.3 as

$$\begin{aligned} \mathbf{y}_t &= A_t \mathbf{y}_{t-1} + B_t \mathbf{u}_{t-1} + \boldsymbol{\epsilon} \\ \mathbf{z}_t &= C_t \mathbf{y}_t + \boldsymbol{\zeta}, \end{aligned} \tag{5.2.1}$$

where

$$\begin{aligned} \boldsymbol{\epsilon} &= \mathcal{N}(\mathbf{0}; R) \\ \boldsymbol{\zeta} &= \mathcal{N}(\mathbf{0}; Q). \end{aligned} \tag{5.2.2}$$

These equations can be used to calculate the posterior distribution [30],

$$p(\mathbf{y}_t | \mathbf{u}_{1:t}, \mathbf{z}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t; \Sigma_t). \tag{5.2.3}$$

The rigid body motion models and measurement models of the system are, however, non-linear. Therefore, a general non-linear description is used for the motion and measurement models $\mathbf{g}$ and $\mathbf{h}$ respectively,

$$\mathbf{y}_t = \mathbf{g}(\mathbf{y}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon} \tag{5.2.4}$$

$$\mathbf{z}_t = \mathbf{h}(\mathbf{y}_t) + \boldsymbol{\zeta}. \tag{5.2.5}$$

The motion and measurement functions, $\mathbf{g}$ and $\mathbf{h}$, are non-linear vector functions. These are linearised to enable the use of the Kalman filter equations. The non-linear vector function, $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_2(\mathbf{x}), f_m(\mathbf{x})]^T$, is linearised around its mean value, $\boldsymbol{\mu}$, using a Taylor series expansion,

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\boldsymbol{\mu}) + \mathbf{f}^{'}(\boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu}), \tag{5.2.6}$$

where

$$\begin{aligned} \mathbf{f}^{'}(\mathbf{x}) = F(\mathbf{x}) &= \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \\ &= \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{m \times n} . \end{aligned} \tag{5.2.7}$$

$F(\mathbf{x})$ is referred to as the Jacobian matrix. Using this linearisation, the vector functions, $\mathbf{g}$ and $\mathbf{h}$, are approximated as,

$$\mathbf{g}(\mathbf{y}_{t-1}, \mathbf{u}_t) \approx \mathbf{g}(\boldsymbol{\mu}_{t-1}, \mathbf{u}_t) + G_t(\mathbf{y}_{t-1} - \boldsymbol{\mu}_{t-1}) \tag{5.2.8}$$

$$\mathbf{h}(\mathbf{y}_t) \approx \mathbf{h}(\boldsymbol{\mu}_t) + H_t(\mathbf{y}_t - \boldsymbol{\mu}_t), \tag{5.2.9}$$

where G_t and H_t are the Jacobian matrices of $\mathbf{g}$ and $\mathbf{h}$, respectively. The linearisation leads to the approximate distribution

$$p(\mathbf{y}_t | \mathbf{u}_{1:t}, \mathbf{z}_{1:t}) \approx \mathcal{N}(\boldsymbol{\mu}_t; \Sigma_t). \tag{5.2.10}$$

With this in mind, the state vector can now be modelled along with the non-linear motion and measurement models.

5.3 System Modelling

Figure 5.1 shows the body-fixed frame of the target, $\mathcal{B}$, and its pose relative to the sensor coordinate system, $\mathcal{C}$. The CRF is assumed to be stationary relative to inertial space. Therefore, all motion of the target is described relative to $\mathcal{C}$. The orientation of the target is described by the quaternion pose vector $\mathbf{q}_{\mathcal{C}/\mathcal{B}}$. The target rotates around its primary axes with angular rates $\boldsymbol{\omega}_{\mathcal{B}} = [\omega_x, \omega_y, \omega_z]^T$. The rates of the target as observed from the sensor are denoted by $\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}$. The current displacement between the target and sensor is denoted by $\mathbf{d}_{\mathcal{C}/\mathcal{B}}$ and the linear velocity of the target is $\mathbf{v}_{\mathcal{C}/\mathcal{B}}$. The target pose vector, $\mathbf{x}_t$, is expressed as,

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{q}_{\mathcal{C}/\mathcal{B}} & \boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}} & \mathbf{d}_{\mathcal{C}/\mathcal{B}} & \mathbf{v}_{\mathcal{C}/\mathcal{B}} \end{bmatrix}^T . \tag{5.3.1}$$

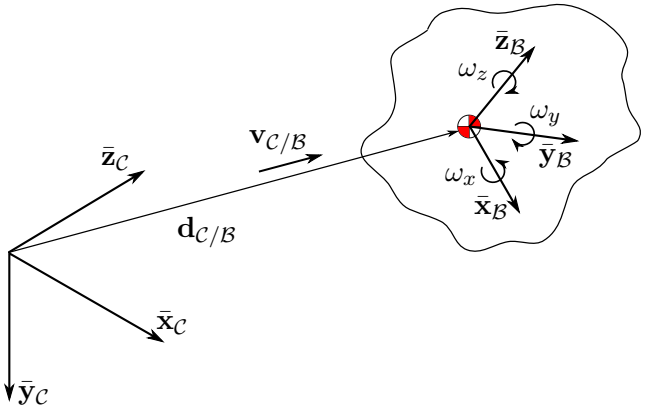


Figure 5.1 – Target reference frame relative to the CRF.

5.3.1 Motion Model

The rotation of the target is non-linear and the propagation of the target motion from one timestep to the next when using Newton-Euler coupling can be described by the vector function, $\mathbf{g}$,

$$\mathbf{x}_t = \mathbf{g}(\mathbf{x}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon}, \tag{5.3.2}$$

which is expanded to

$$\mathbf{x}_t = \begin{bmatrix} q_{t,1} \\ q_{t,2} \\ q_{t,3} \\ q_{t,4} \\ w_{t,x} \\ w_{t,y} \\ w_{t,z} \\ d_{t,x} \\ d_{t,y} \\ d_{t,z} \\ v_{t,x} \\ v_{t,y} \\ v_{t,z} \end{bmatrix} = \mathbf{x}_{t-1} + \begin{bmatrix} \frac{1}{2}(-\omega_{t-1,x}q_{t-1,2} - \omega_{t-1,y}q_{t-1,3} - \omega_{t-1,z}q_{t-1,4}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,1} - \omega_{t-1,y}q_{t-1,4} + \omega_{t-1,z}q_{t-1,3}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,4} + \omega_{t-1,y}q_{t-1,1} - \omega_{t-1,z}q_{t-1,2}) \\ \frac{1}{2}(-\omega_{t-1,x}q_{t-1,3} + \omega_{t-1,y}q_{t-1,2} + \omega_{t-1,z}q_{t-1,1}) \\ ((I_{yy} - I_{zz})\omega_{t-1,y}\omega_{t-1,z})/I_{xx} \\ ((I_{zz} - I_{xx})\omega_{t-1,x}\omega_{t-1,z})/I_{yy} \\ ((I_{xx} - I_{yy})\omega_{t-1,x}\omega_{t-1,y})/I_{zz} \\ v_{t-1,x} \\ v_{t-1,y} \\ v_{t-1,z} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta t + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ T_x/I_{xx} \\ T_y/I_{yy} \\ T_z/I_{zz} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.3.3}$$

The body-fixed axes of the target are chosen to coincide with its principal axes of inertia. The principal moments of inertia are given by

$$\mathbf{I}_{\mathcal{B}} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}. \tag{5.3.4}$$

External torques, T_x, T_y and T_z, are assumed to be zero and there is thus no control input, $\mathbf{u}_t$, in Equation 5.3.5. This makes sense since the target is non-cooperative and the motion of the target is estimated without any intrinsic knowledge of the moments of inertia. It is not possible to get an indication of $\mathbf{I}_{\mathcal{B}}$, because no prior information regarding the shape or mass distribution of the target is available. Therefore, the inertial coupling between body rates is removed and the motion model is reduced to

$$\mathbf{x}_t = \mathbf{g}(\mathbf{x}_{t-1}), \tag{5.3.5}$$

where

$$\mathbf{g}(\mathbf{x}_{t-1}) = \mathbf{x}_{t-1} + \begin{bmatrix} \frac{1}{2}(-\omega_{t-1,x}q_{t-1,2} - \omega_{t-1,y}q_{t-1,3} - \omega_{t-1,z}q_{t-1,4}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,1} - \omega_{t-1,y}q_{t-1,4} + \omega_{t-1,z}q_{t-1,3}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,4} + \omega_{t-1,y}q_{t-1,1} - \omega_{t-1,z}q_{t-1,2}) \\ \frac{1}{2}(-\omega_{t-1,x}q_{t-1,3} + \omega_{t-1,y}q_{t-1,2} + \omega_{t-1,z}q_{t-1,1}) \\ 0 \\ 0 \\ 0 \\ v_{t-1,x} \\ v_{t-1,y} \\ v_{t-1,z} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta t. \tag{5.3.6}$$

Excluding the body-rate coupling from the motion model may lead to a phase delay or inaccuracies in the estimation of the angular rate values, since the rates will solely be dependent on the corrections brought about by the measurement update.

5.3.2 Measurement Model

Chapter 4 discussed how feature-based measurements are extracted from stereo camera images. The measurements in world coordinates are now described relative to the target reference frame. The combined state vector is comprised of the target pose vector, $\mathbf{x}_t$ and a map of features, $\mathbf{m}_t$, and is given by

$$\mathbf{y}_t = \begin{bmatrix} \mathbf{x}_t & m_{1,r} & m_{1,\psi} & m_{1,\theta} & \cdots & m_{N,r} & m_{N,\psi} & m_{N,\theta} \end{bmatrix}^T. \tag{5.3.7}$$

The position of the n-th feature on the target for every feature $n = 1, ..., N$ is denoted by $m_{n,r}$, $m_{n,\psi}$ and $m_{n,\theta}$. The dimension of the combined state vector is $13 + 3N$, where N is the number of features observed up until time t. Features are described in the target coordinate system, $\mathcal{B}$. The position of a feature on the target surface is shown in Figure 5.2.

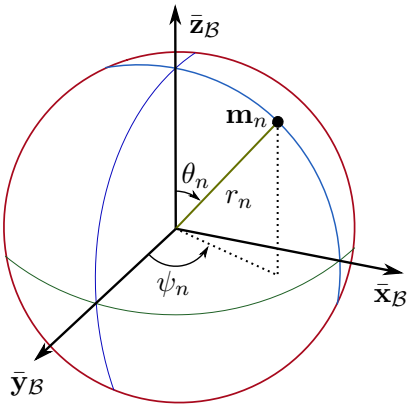


Figure 5.2 – A feature described in the target reference frame.

The position of $\mathbf{m}_n$ in Cartesian coordinates is expressed as

$$\begin{aligned} x_n &= r_n \sin\psi_n \cos\theta_n \\ y_n &= r_n \sin\psi_n \sin\theta_n \\ z_n &= r_n \cos\psi_n. \end{aligned} \tag{5.3.8}$$

Features are observed on the surface of the target and are described in spherical coordinates relative to its centre of rotation. The target consists of N features and at a given time t, K features are observed by the sensor. Each observation of feature $\mathbf{m}_n$ in the body-fixed reference frame $\mathcal{B}$, results in a measurement $\mathbf{z}_{t,k}$ and is denoted as

$$\mathbf{m}_n = \begin{bmatrix} r_n \\ \psi_n \\ \theta_n \end{bmatrix} \qquad \text{and} \qquad \mathbf{z}_{t,k} = \begin{bmatrix} x_{t,k} \\ y_{t,k} \\ z_{t,k} \end{bmatrix}. \tag{5.3.9}$$

The measurements are combined to form the measurement set,

$$\mathbf{z}_t = \begin{bmatrix} \mathbf{z}_{t,1}^T \\ \vdots \\ \mathbf{z}_{t,K}^T \end{bmatrix}, \tag{5.3.10}$$

where every measurement $\mathbf{z}_{t,k}$, has a correspondence $c_{t,k} = n$, to label each feature according to the feature map. The sensor observes a feature in Cartesian coordinates and the features in the CRF are expressed as,

$$\begin{aligned}\mathbf{z}_{t,k} &= \begin{bmatrix} x_{t,k} \\ y_{t,k} \\ z_{t,k} \end{bmatrix} = \mathbf{h}(\mathbf{x}_t, \mathbf{m}_{n,t}) + \boldsymbol{\zeta} \\ &= \mathbf{A}_{\mathcal{C},t}^{\mathcal{B}} \begin{bmatrix} r_n \sin\psi_n \cos\theta_n \\ r_n \sin\psi_n \sin\theta_n \\ r_n \cos\psi_n \end{bmatrix} + \begin{bmatrix} d_{t,x} \\ d_{t,y} \\ d_{t,z} \end{bmatrix} + \boldsymbol{\zeta}, \end{aligned} \tag{5.3.11}$$

where the DCM, $\mathbf{A}_{\mathcal{C},t}^{\mathcal{B}}$, describes the orientation between the target and the sensor at time t. The inverse of the measurement function is written as,

$$\begin{aligned}\mathbf{m}_{n,t} &= \begin{bmatrix} r_n \\ \psi_n \\ \theta_n \end{bmatrix} = \mathbf{h}^{-1}(\mathbf{x}_t, \mathbf{z}_{t,k}) \\ &= \begin{bmatrix} \sqrt{x_n + y_n + z_n} \\ \arctan 2(y_n, x_n) \\ \arccos(z_n / \sqrt{x_n + y_n + z_n}) \end{bmatrix}, \end{aligned} \tag{5.3.12}$$

where

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = (\mathbf{A}_{\mathcal{C},t}^{\mathcal{B}})^T \begin{bmatrix} x_{t,k} - d_{t,x} \\ y_{t,k} - d_{t,y} \\ z_{t,k} - d_{t,z} \end{bmatrix}. \tag{5.3.13}$$

It is useful to define this inverse function, since it is used to add new features to the state vector. The tracking algorithm using the EKF is derived using the motion and measurement models.

5.4 Target Tracking

The target pose and feature positions are tracked similar to the EKF-SLAM algorithm proposed by Thrun *et al.* [30] and is described in Algorithm 3. The algorithm consists of two parts: the motion update in lines 1 to 3 and the measurement update in lines 4 to 15. The motion update calculates a predicted distribution over the current state given the previous state vector. It is expected that the uncertainty in target pose, $\Sigma_{\mathbf{x}}$, will increase in this step, since no new information is used. The measurement update calculates the posterior distribution over the target pose vector and the feature positions according to

$$p(\mathbf{y}_t|\mathbf{z}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t; \Sigma_t) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_{t,\mathbf{x}} \\ \boldsymbol{\mu}_{t,\mathbf{m}} \end{bmatrix}; \begin{bmatrix} \Sigma_{t,\mathbf{x}} & \cdots & \cdots & \cdots \\ \vdots & \ddots & \cdots & \cdots \\ \vdots & \vdots & \Sigma_{t,\mathbf{m}} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \right), \tag{5.4.1}$$

which is divided between the target pose vector and the feature map vector. It is expected that the uncertainty of the target pose $\Sigma_{\mathbf{x}}$, as well as the uncertainty of feature positions, $\Sigma_{\mathbf{m}}$, will decrease after a measurement update takes place.

Algorithm 3 Extended Kalman Filter Target Tracking

1: **function** STEPEKF($\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{z}_t$)
2: $\quad \bar{\boldsymbol{\mu}}_t = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\mu}_{t-1,\mathbf{m}} \end{bmatrix} + \begin{bmatrix} \mathbf{g}(\boldsymbol{\mu}_{t-1}) \\ \mathbf{0} \end{bmatrix}$
3: $\quad \bar{\Sigma}_t = \begin{bmatrix} G_t & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix} \Sigma_{t-1} \begin{bmatrix} G_t & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix}^T + \begin{bmatrix} R & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$
4: $\quad$ **for** all observed features $\mathbf{z}_{t,k}$ **do**
5: $\quad\quad j = c_{t,k}$
6: $\quad\quad$ **if** feature j never observed before **then**
7: $\quad\quad\quad \mathbf{m}_{t,j} = \mathbf{h}^{-1}(\mathbf{x}_t, \mathbf{z}_{t,k})$
8: $\quad\quad\quad \bar{\Sigma}_t = \begin{bmatrix} \bar{\Sigma}_t & \mathbf{0} \\ \mathbf{0} & \infty_{3\times 3} \end{bmatrix}$
9: $\quad\quad$ **end if**
10: $\quad\quad \hat{\mathbf{z}}_{t,k} = \mathbf{h}(\mathbf{x}_t, \mathbf{m}_{t,j})$
11: $\quad\quad H_{t,k} = \begin{bmatrix} H_{t,x} & \mathbf{0} & H_{t,j} & \mathbf{0} \end{bmatrix}$
12: $\quad\quad K_{t,k} = \bar{\Sigma}_t H_{t,k}^T (H_{t,k} \bar{\Sigma}_t H_{t,k}^T + Q)^{-1}$
13: $\quad\quad \bar{\boldsymbol{\mu}}_t = \bar{\boldsymbol{\mu}}_t + K_t^i (\mathbf{z}_{t,k} - \hat{\mathbf{z}}_{t,k})$
14: $\quad\quad \bar{\Sigma}_t = (I - K_{t,k} H_{t,k}) \bar{\Sigma}_t$
15: $\quad$ **end for**
16: $\quad \boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t$
17: $\quad \Sigma_t = \bar{\Sigma}_t$
18: $\quad$ **return** $\boldsymbol{\mu}_t, \Sigma_t$
19: **end function**

The most important steps in Algorithm 3 are discussed for clarification:

- **Line 2:** The motion update is done using Equation 5.3.5 to update the mean values of the state vector.
- **Line 3:** The predicted state covariance, $\bar{\Sigma}_t$ is calculated. The Jacobian matrix of the motion model, G_t is divided into groups to simplify notation,

$$G_t = \begin{bmatrix} \frac{1}{2} G_q & \mathbf{0} \\ 0 & \mathbf{0} \\ \mathbf{0} & G_d \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{5.4.2}$$

  where G_q represents the quaternion attitude part and G_d the translation. These matrices are expressed as:

$$G_q = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z & -q_2 & -q_3 & -q_4 \\ \omega_x & 0 & \omega_z & -\omega_y & q_1 & -q_4 & -q_3 \\ \omega_y & -\omega_z & 0 & \omega_x & q_4 & q_1 & -q_2 \\ \omega_z & \omega_y & -w_x & 0 & -q_3 & q_2 & q_1 \end{bmatrix}$$

and

$$G_d = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

- **Lines 4-5:** Every measurement is considered individually and the measurement update is repeated for all the features observed at time t. The index j is used to indicate the position of each feature in the state vector, and k is the position of the feature in the measurement vector given by the correspondence vector, $\mathbf{c}$.

- **Lines 6-8:** Features that are observed for the first time are added to the state vector using the inverse measurement function in Equation 5.3.12. A theoretical value of infinity is used to initialise the uncertainty of the newly added features.

- **Line 10:** The position of the feature in the CRF is calculated. The current estimate of the feature is transformed using the measurement model in Equation 5.3.11. The measurement and predicted values are compared in line 13 to calculate the innovation, $\mathbf{z}_{t,k} - \hat{\mathbf{z}}_{t,k}$.

- **Line 11:** The Jacobian for the measurement model is split in two parts. One part represents the target pose vector, $\mathbf{x}_t$ and the other represents the measurement at a position, j, in the state vector and covariance matrix, Σ_t. The Jacobian for the target states part is derived as

$$H_{t,x} = \frac{1}{2} \begin{bmatrix} q_1x - q_4y + q_3z & q_4x + q_1y - q_2z & -q_3x + q_2y + q_1z \\ q_2x + q_3y + q_4z & q_3x - q_2y - q_1z & q_4x + q_1y - q_2z \\ -q_3x + q_2y + q_1z & q_2x + q_3y + q_4z & -q_2x + q_4y - q_3z \\ -q_4x - q_1y + q_2z & q_1x - q_4y + q_3z & q_2x + q_3y + q_4z \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}^T,$$

where $x = r_j \sin(\psi_j)\cos(\theta_j)$, $y = r_j \sin(\psi_j)\sin(\theta_j)$ and $z = r_j \cos(\psi_j)$. The notation of measurement Jacobian for the feature part is simplified by defining the following variables:

$$
\begin{aligned}
k_1 &= q_1^2 + q_2^2 - q_3^2 - q_4^2 \\
k_2 &= (q_2 q_3 - q_1 q_4) \\
k_3 &= (q_2 q_4 + q_1 q_3) \\
k_4 &= (q_1 q_4 + q_2 q_3) \\
k_5 &= q_1^2 - q_2^2 + q_3^2 - q_4^2 \\
k_6 &= (q_3 q_4 - q_1 q_2) \\
k_7 &= (q_2 q_4 - q_1 q_3) \\
k_8 &= (q_1 q_2 + q_3 q_4) \\
k_9 &= q_1^2 - q_2^2 - q_3^2 + q_4^2 \\
\alpha_1 &= \sin(\psi)\cos(\theta) \\
\alpha_2 &= \sin(\psi)\sin(\theta) \\
\alpha_3 &= \cos(\psi)\cos(\theta) \\
\alpha_4 &= \cos(\psi)\sin(\theta)
\end{aligned}
$$

$H_{t,j}$ is now described by:

$$
\begin{bmatrix}
\alpha_1 k_1 + 2\alpha_2 k_2 + 2\cos(\psi)k_3 & 2\alpha_1 k_4 + \alpha_2 k_5 + \cos(\psi)k_6 & 2\alpha_1 k_7 + \alpha_1 k_8 + \cos(\psi)k_9 \\
-r\alpha_2 k_1 + 2r\alpha_1 k_2 & -2r\alpha_2 k_4 + r\alpha_1 k_5 & -2r\alpha_2 k_7 + 2r\alpha_1 k_8 \\
r\alpha_3 k_1 + r\alpha_4 k_2 - 2r\sin(\psi)k_3 & 2r\alpha_3 k_4 + r\alpha_4 k_5 - 2r\sin(\psi)k_6 & 2r\alpha_3 k_7 + 2r\alpha_4 k_8 - r\sin(\psi)k_9
\end{bmatrix}^T . \tag{5.4.3}
$$

- **Lines 12-14:** The Kalman gain is calculated and used to update the state vector with the error between the estimated feature position and measured position. The state covariance is also updated using the measurement Jacobian and the gain, $K_{t,k}$.
- **Line 16:** When all of the measurements are processed in the current timestep, the updated mean and covariance is returned.

5.4.1 Feature Pruning

It is necessary to often remove unused features from the state vector to reduce computations. A tracked feature is pruned if there are no new measurement updates of that feature for 5 consecutive timesteps. This is likely caused by features being occluded from the field of view and maintaining them does not provide any new information. As mentioned earlier, the state covariance matrix has dimensions $[13 + 3N \times 13 + 3N]$ with N the number of tracked features. This can quickly grow to become unmanageable when a large amount of features are tracked and may lead to numerical errors. Therefore a feature is removed from the state vector and the feature is marginalised from Σ_t. These values are stored offline and are used to perform target reconstruction.

5.4.2 Initial Values

Since the relative and not the absolute pose of the target is tracked, the initial attitude of the target is aligned to the CRF. Therefore, there is no relative attitude difference between the target and sensor coordinate systems at time $t = 0$.

A value for the initial displacement between the coordinate systems is calculated using the mean position of all the features observed at the first timestep. The motion of a feature on a large rotating object may be observed to be the same as the motion of a feature on a small target that is translating rather than rotating. Providing the system with an approximate displacement thus helps the estimator to distinguish between these two cases.

The sensor only observes exposed target surfaces facing towards the chaser. Calculating a mean position using the exposed features may lead to a bias in depth calculation. The uncertainty over the depth initial displacement is therefore larger than for the horizontal and vertical positions.

The addition of an offset parameter to the target state vector could possibly account for large errors made in the initial displacement assignment. The offset will be a vector describing the position between the true centre of rotation and the initial position of the estimated body reference frame. However, early simulation results showed that this was not necessary when the whole target was in the field of view at the start of the estimation. This is a valid assumption, since a chaser satellite will in most cases rendezvous with the target from a distance large enough to observe the whole target.

5.5 Simulation

A controlled simulation environment is designed to generate observation sets of a target moving in the sensor reference space. The measurements generated by the simulation is similar to what can be expected from the stereo-camera sensor discussed in Chapter 4. Simulation provides ground truth data of the target's motion and the estimation results are directly compared to assess the functionality and performance of the estimator design. The simulation also allows the estimation of Newton-Euler rotations in a gravity- and friction-free environment.

The target is created as a model of a satellite with solar panels, represented by a cloud of points on the surfaces of two cylinders and two rectangular cuboids. Features are chosen by randomly sampling a specified number of points on the target. This prevents the feature positions from influencing the behaviour of the system. Measurement sets are generated by propagating these features over time using the three principal body rates and linear velocity vector of the target. Figure 5.3 shows the sparse point cloud that represents the target. The principal axes of inertia are indicated in blue and a sample of 40 randomly chosen features are shown in red. Occlusion handling is also tested in the simulation. Occlusion planes are defined as a function of the target position relative to the sensor field of view (FOV) and the target's size. If a feature is behind the occlusion plane, then it is removed from the measurement set for that timestep.

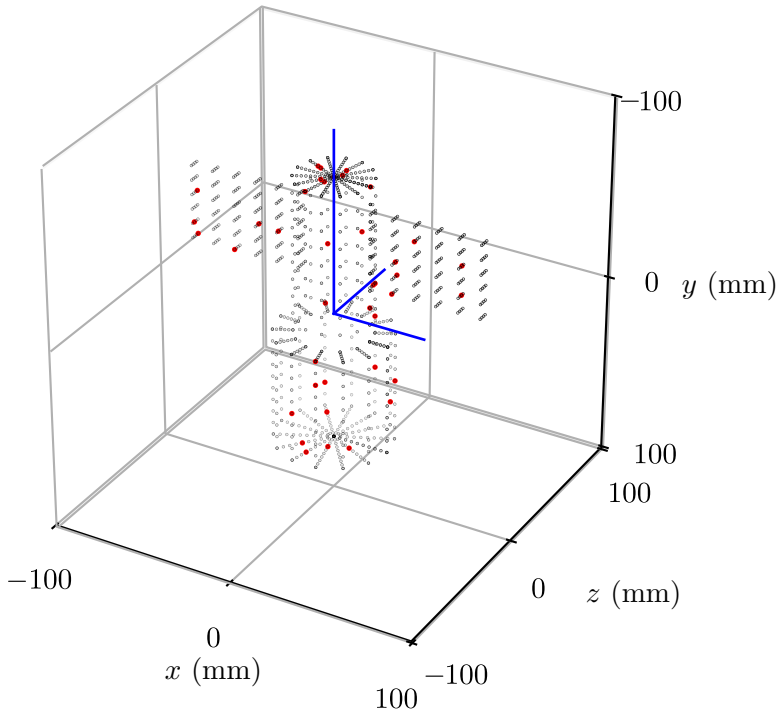


Figure 5.3 – Simulated target with randomly selected features shown in red.

The target rotates and translates relative to the stationary CRF. The target's initial angular velocity vector is indicated by $\boldsymbol{\omega}_{\mathcal{B}} = [\omega_x, \omega_y, \omega_z]^T$ and the initial displacement is denoted as $\mathbf{d}_{\mathcal{C}/\mathcal{B}} = [d_x, d_y, d_z]^T$. The target translates sinusoidally with an initial velocity vector $\mathbf{v}_{\mathcal{B}} = [v_x, v_y, v_z]^T$. Table 5.1 provides the values of the simulation parameters. The angular rate values will allow at least one full rotation to be completed in the allowed time, thus exposing all the selected features to the sensor.

$\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}$ (rad/s)	$\mathbf{v}_{\mathcal{B}}$ (m/s)	$\mathbf{d}_{\mathcal{C}/B}$ (m)	Timesteps	Step size (s)	Features
$\omega_1 = 0.3$ $\omega_2 = 0.5$ $\omega_3 = -0.4$	$v_x = 0.1$ $v_y = -0.1$ $v_z = 0.2$	$d_x = 2$ $d_y = 1$ $d_z = 5$	400	0.05	40

Table 5.1 – Simulation parameters

Using a step size of 0.05 s relates to a sensor framerate of 20 Hz. The simulation is carried out for 20 seconds, resulting in 400 total steps. Once all measurement inputs have been generated, they are provided to the EKF that solves the pose estimation problem. It estimates the distribution of the target state vector along with the distribution of the feature positions on the target.

The simulation results shown in this section are compared to ground truth data. Figure 5.4 shows the estimated values of the quaternion attitude vector elements with the ground truth values in dotted black lines. There is no initial difference between the estimated and ground truth values, since the initial attitude is chosen to match the sensor's. It is evident that the estimator is fully capable of tracking the quaternion attitude. Although the attitude is tracked fairly well, there exists a phase delay in the estimated angular velocity vector $\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}$ in Figure 5.5. The target is rotating according to Newton-Euler dynamics and a cross-coupling exists between the body rates. This coupling is not modelled in the kinematic motion model. An accurate indication of the target's inertia is required to model the cross-coupling and this is not possible to calculate with sparse features.

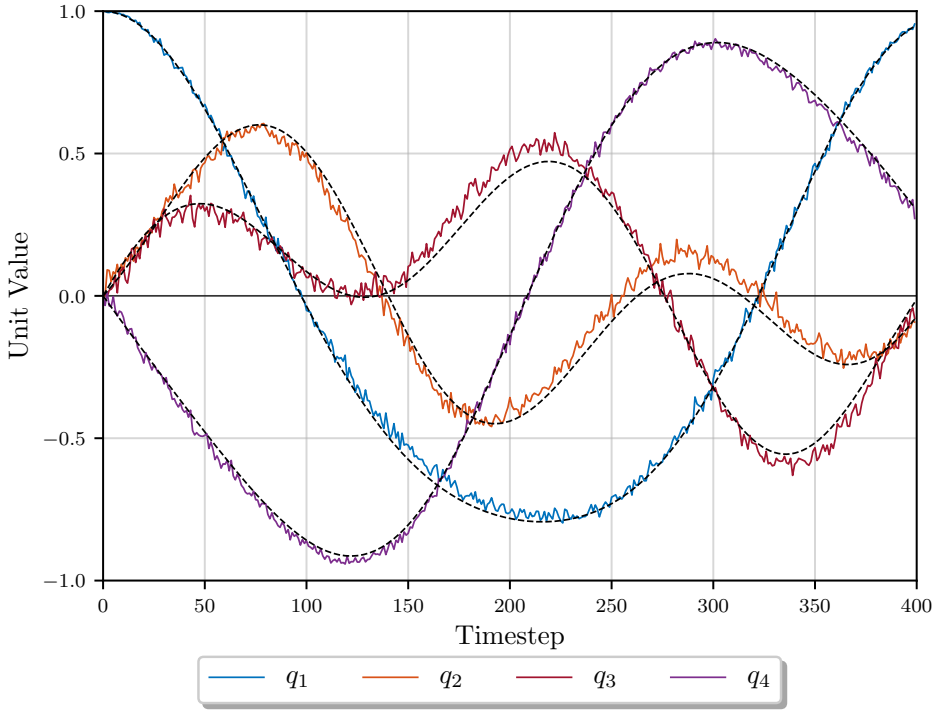


Figure 5.4 – Estimated quaternion attitude from simulated measurements. Ground truth values are indicated by the dotted lines.

Although the angular rate estimation produces erroneous results, it does not affect the estimation of the other states to a large extent. It can be concluded that the measurement model has a larger influence than the incomplete motion model on the system states. Section 5.5.1 investigates the improvement of adding rotational dynamics to the motion model.

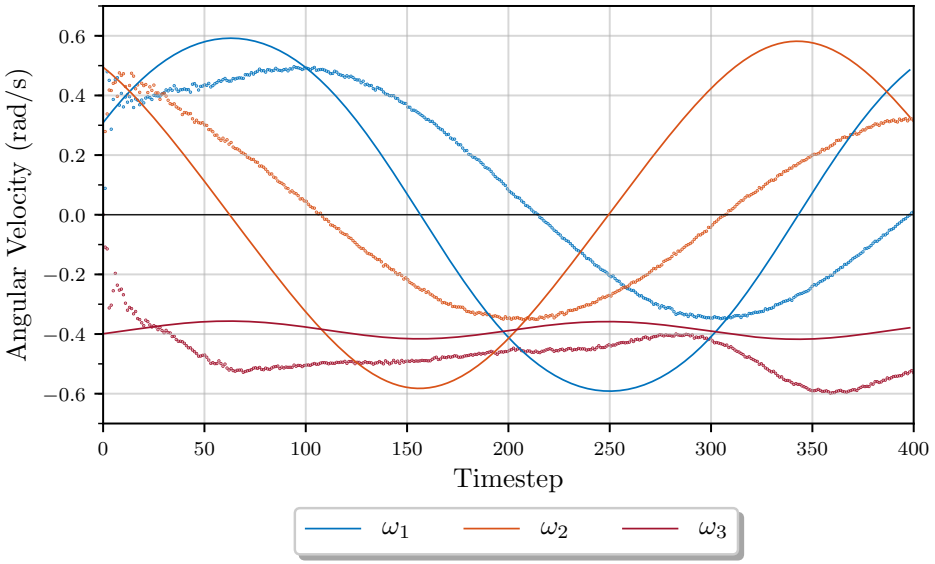


Figure 5.5 – Estimated angular velocities from simulated measurements.

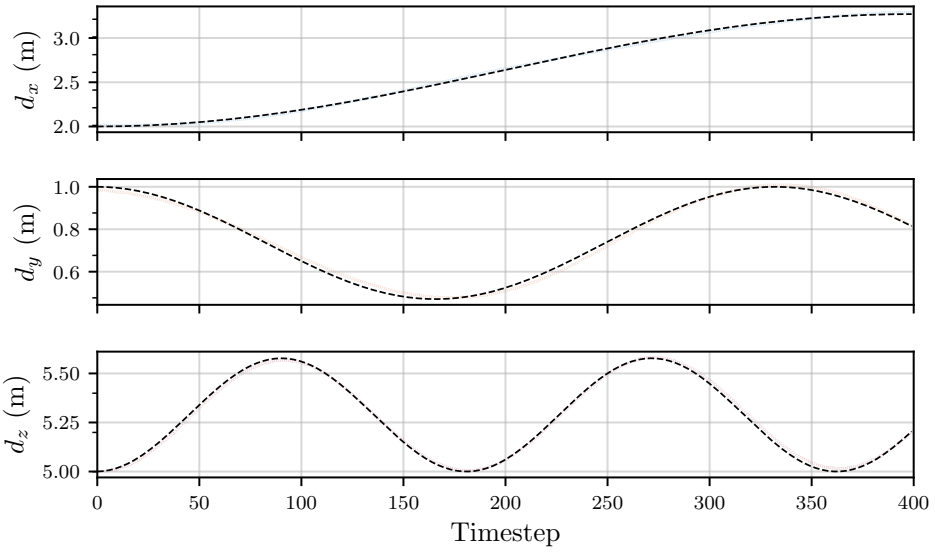


Figure 5.6 – Relative displacement between the sensor and the target. Ground truth values are indicated by the dotted lines.

The relative displacement between the sensor and target is shown in Figure 5.6. The translation estimates are satisfactory and the choosing of the initial displacement vector results in a very small difference between the estimated and ground truth values. The velocity vector of the target is shown in Figure 5.7. The result is more

noisy than the displacement estimate, since the velocity is only updated from the derivative of the displacements.

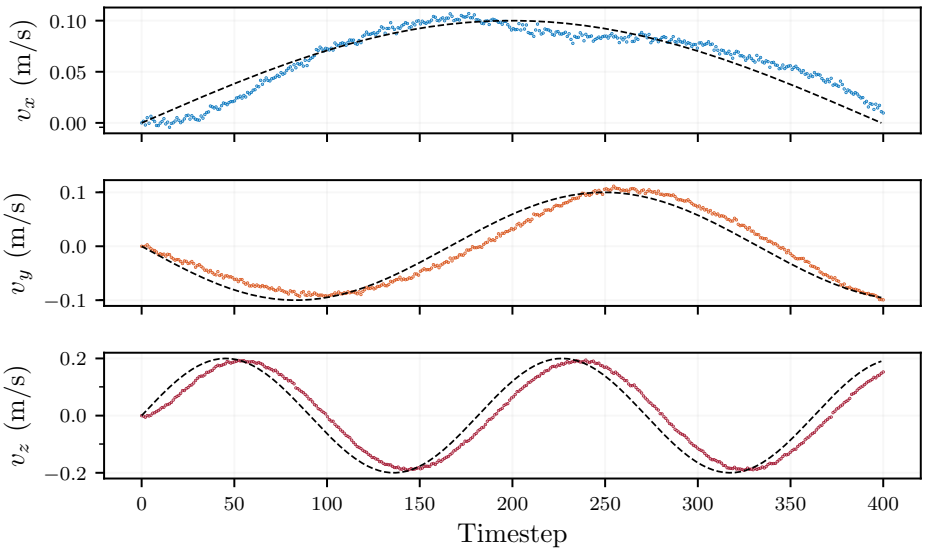


Figure 5.7 – Linear velocities between the sensor and the target. Ground truth values are indicated by the dotted black lines.

5.5.1 Addition of Newton-Euler Dynamics in Motion Model

A case where the principal moments of inertia are known, is investigated. The inertia tensor is calculated using the sparse point cloud of the target that was shown in Figure 5.3. The geometric centroid, $\mathbf{r}_m$, in the target reference frame is expressed as

$$\mathbf{r}_m = \frac{\sum \mathbf{r}_i}{n}, \tag{5.5.1}$$

where n is the number of points in the point cloud and $\mathbf{r}_i$ is the position of the i^{th} point in the target's coordinate system. It is assumed that each point has the same weight and the principal second moments about the points are calculated as follows,

$$\begin{aligned} I_{xx} &= \sum n_i \left((r_i^y - r_m^y)^2 + (r_i^z - r_m^z)^2 \right) \\ I_{yy} &= \sum n_i \left((r_i^x - r_m^x)^2 + (r_i^z - r_m^z)^2 \right) \\ I_{zz} &= \sum n_i \left((r_i^x - r_m^x)^2 + (r_i^y - r_m^y)^2 \right). \end{aligned}$$

The motion model shown in Equation 5.3.5 is changed to make use of the new moment of inertia information. It is expected that the tracking of the body rates

will improve greatly since the rates are updated using more information than just the derivative of the attitude quaternion. The motion model using Newton-Euler dynamics is expressed as

$$\mathbf{g}(\mathbf{x}_{t-1}) = \mathbf{x}_{t-1} + \begin{bmatrix} \frac{1}{2}(-\omega_{t-1,x}q_{t-1,2} - \omega_{t-1,y}q_{t-1,3} - \omega_{t-1,z}q_{t-1,4}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,1} - \omega_{t-1,y}q_{t-1,4} + \omega_{t-1,z}q_{t-1,3}) \\ \frac{1}{2}(\omega_{t-1,x}q_{t-1,4} + \omega_{t-1,y}q_{t-1,1} - \omega_{t-1,z}q_{t-1,2}) \\ \frac{1}{2}(-\omega_{t-1,x}q_{t-1,3} + \omega_{t-1,y}q_{t-1,2} + \omega_{t-1,z}q_{t-1,1}) \\ ((I_{yy} - I_{zz})\omega_{t-1,y}\omega_{t-1,z})/I_{xx} \\ ((I_{zz} - I_{xx})\omega_{t-1,x}\omega_{t-1,z})/I_{yy} \\ ((I_{xx} - I_{yy})\omega_{t-1,x}\omega_{t-1,y})/I_{zz} \\ v_{t-1,x} \\ v_{t-1,y} \\ v_{t-1,z} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta t. \tag{5.5.2}$$

It is logical that the Jacobian also has to change to account for the new information. A new matrix, G_ω is added to the motion update Jacobian in Equation 5.4.2,

$$G_t = \begin{bmatrix} \frac{1}{2}G_q & \mathbf{0} \\ G_\omega & \mathbf{0} \\ \mathbf{0} & G_d \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.5.3}$$

where

$$G_\omega = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \omega_z(I_y - I_z)/I_x & \omega_y(I_y - I_z)/I_x \\ 0 & 0 & 0 & 0 & \omega_z(I_z - I_x)/I_y & 0 & \omega_x(I_z - I_x)/I_y \\ 0 & 0 & 0 & 0 & \omega_y(I_x - I_y)/I_z & \omega_x(I_x - I_y)/I_z & 0 \end{bmatrix}_t \tag{5.5.4}$$

A simulation is performed with the parameters in Table 5.1, but this time the target's principal moments of inertia are known. The improvement in angular rate estimation when the motion model includes cross-coupling between angular velocities, can be clearly seen in Figure 5.8.

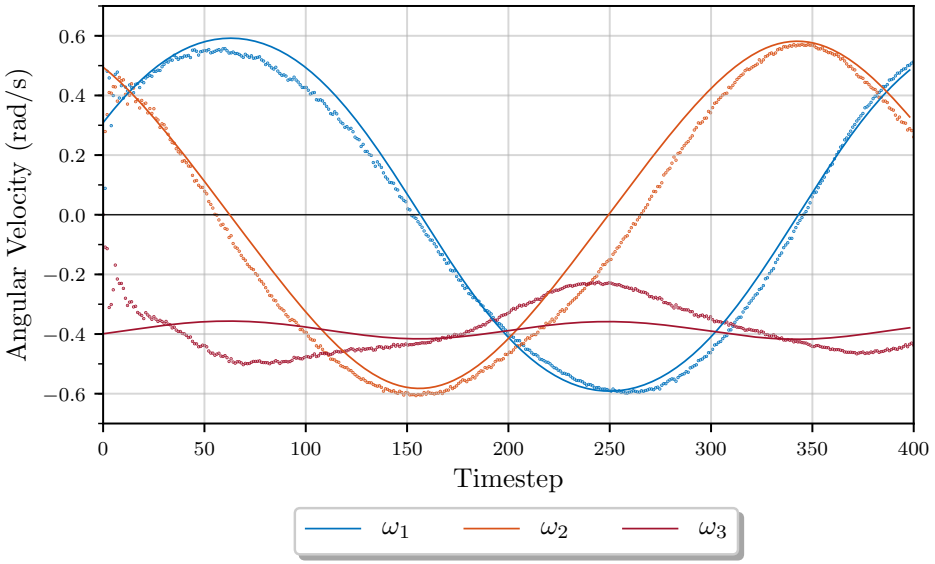


Figure 5.8 – Angular velocities estimated using knowledge of target moments of inertia. Solid lines indicate ground truth, with estimated values as dotted lines in matching colour.

5.6 Practical Considerations

There are some practical constraints to the estimator, especially with regards to the nature of the measurements that can be expected using the stereo-camera sensor. The minimum number of required features that have to be tracked is established, along with the influence of measurement outliers.

5.6.1 Number of Features

The case where all features are always visible is explored to obtain a theoretic minimum baseline number of tractable features that will still deliver acceptable performance. For this test, occlusion planes are removed to generate simulated data sets. The number of tracked features range from two to six for each set and the results are plotted against each other for comparison.

The estimated target attitude is shown in Figure 5.9. It is clear from the figure that the performance is significantly reduced when there are four or less tracked features. The minimum threshold will increase when using data with occlusions. Features may only be tracked for a few timesteps or in some cases might just appear once and then move out of the field of view again. A lack of features may originate from blurry images or if there are shadows on the surfaces of the target. The target may also be positioned at such an angle that only a small part of it is visible to the camera.

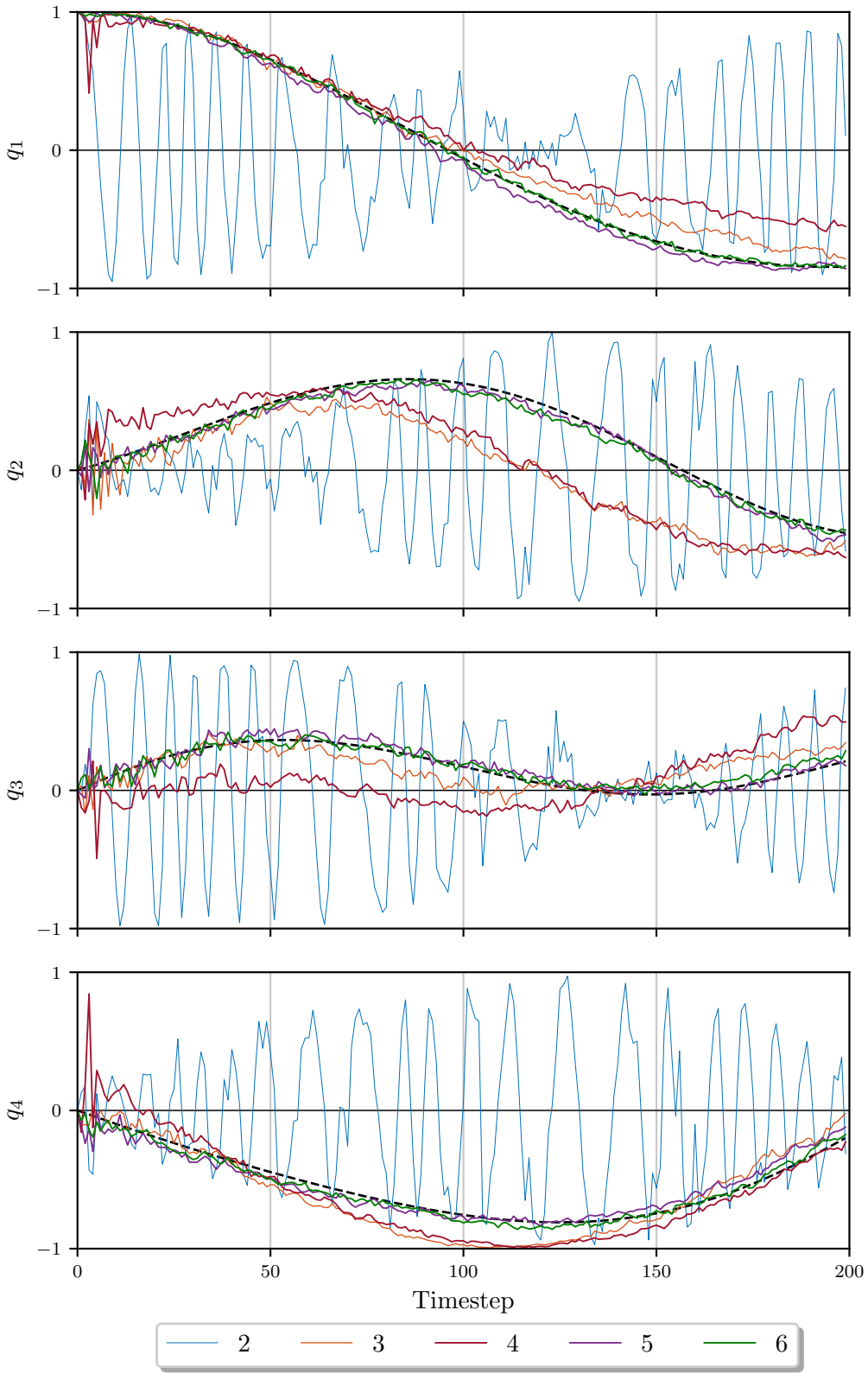


Figure 5.9 – Estimated quaternion attitude vector using features that are always visible. The dotted black lines indicate ground truth values. The number of visible features are varied between two and six points.

To test the effect of occlusions, four different data sets are generated, each with a different number of features. The number of total target features are varied between 10 and 40. These features are uniformly sampled from the target point cloud that was shown in Figure 5.3. The number of visible features will vary with time depending on the movement of the target. Figure 5.10 indicates the number of visible features at each timestep for the four cases.

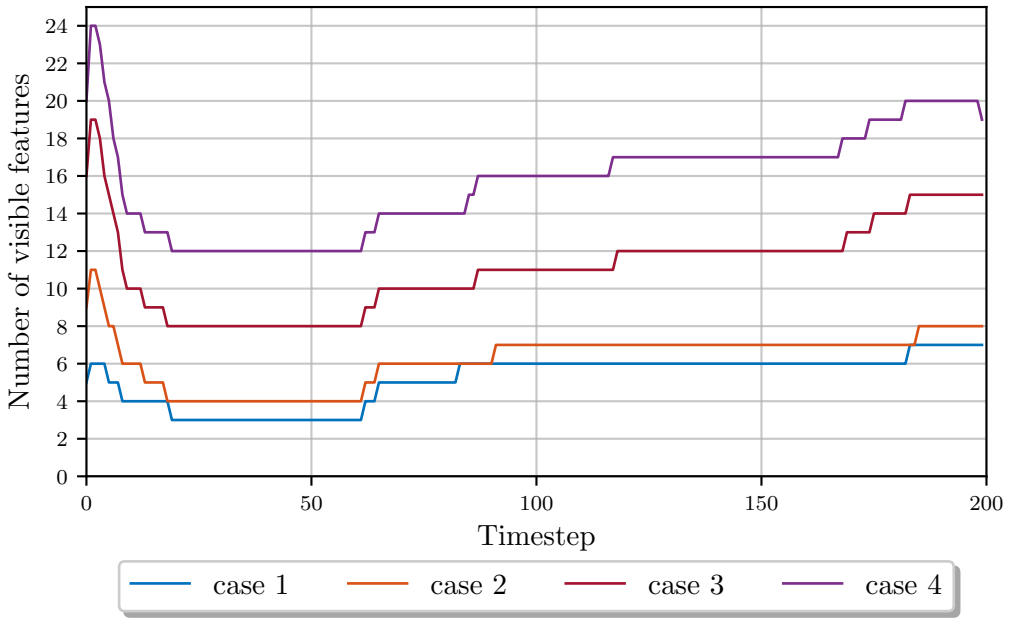


Figure 5.10 – Number of visible points for four different cases. The number of total available features for each dataset is: case 1 = 10, case 2 = 20, case 3 = 30, case 4 = 40.

Figure 5.11 shows the estimated quaternion attitude using occluded features. It is clear for case 1 that the target attitude tracking fails. This is expected since it was proven that the estimator requires at the very minimum 5 features that are always visible to deliver satisfactory performance. There are only 3 to 6 features visible for the most of the simulation in case 1. There is a noticeable performance increase for case 2, although large errors are made at the start of the simulation. There is, however, an increase in visible features in the last half of the simulation and the estimator is able to follow the ground truth, albeit offset errors. Cases 3 and 4 display a remarkable improvement in estimation quality. The number of visible features are always in excess of eight points.

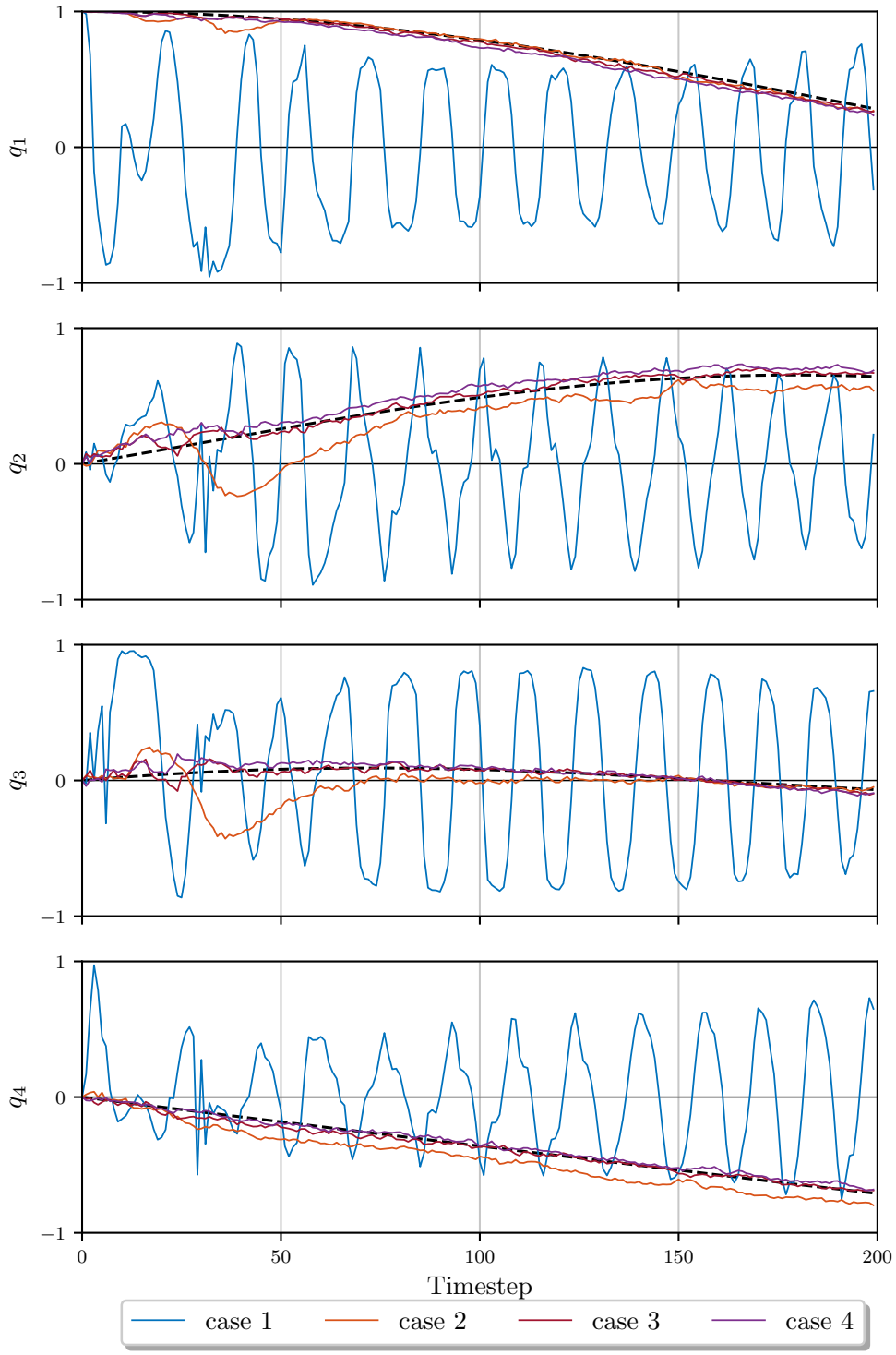


Figure 5.11 – Estimated quaternion attitude vector using occluded features. Ground truth values are indicates using dotted black lines.

5.6.2 Outliers

The effect of outliers on the performance of the system is tested by adding high amplitude noise to some of the measurements. Since outliers will most likely originate from mismatches in the stereo matching algorithm from Section 4.3.7, random values from a uniform distribution with a magnitude 15 times greater than the expected measurement noise are added to the depth of the measurements.

This is a valid way of mimicking outliers since the mismatches will likely be caused by high noise originating from adverse lighting or light reflecting from glossy surfaces creating speckles on the image planes. The mismatches will lead to errors in the depth calculation of the measurements. Unexpected noise may also originate if the heat generated by the cameras and other satellite components are not dissipated efficiently. The number of outliers that are introduced are progressively increased until the influence of the outliers significantly degrades the estimation quality. This test uses a total of 40 available target features; the best case from the tests in the previous section. Table 5.2 indicates the timesteps at which outliers occur and the ratio between outliers and visible features at that timestep.

Timestep #	Outlier to feature ratio
40	15.7%
120	29.4%
200	78.9%

Table 5.2 – Occurrence of outliers in simulated data.

Figure 5.12 shows the results of a test case where outliers were introduced. It displays the quaternion error between the estimated and ground truth values. Only the vector elements are displayed, since the quaternion identity

$$||\mathbf{q}|| = 1, \tag{5.6.1}$$

allows the calculation of the scalar element given the other three values. The quaternion error is calculated as the nett rotation required to transform the estimated attitude to the ground truth attitude. The error shows that an attitude shift occurred at $t = 40$ and $t = 120$ where the outliers were introduced. The attitude remains stable, but with this offset error. The outliers at $t = 200$ cause a divergence in attitude tracking and the estimator is not able to revert back to stable tracking after this. Although it is critical to remove outliers from measurement data for reliable estimation, the ratio of outliers required to diverge estimation is very high.

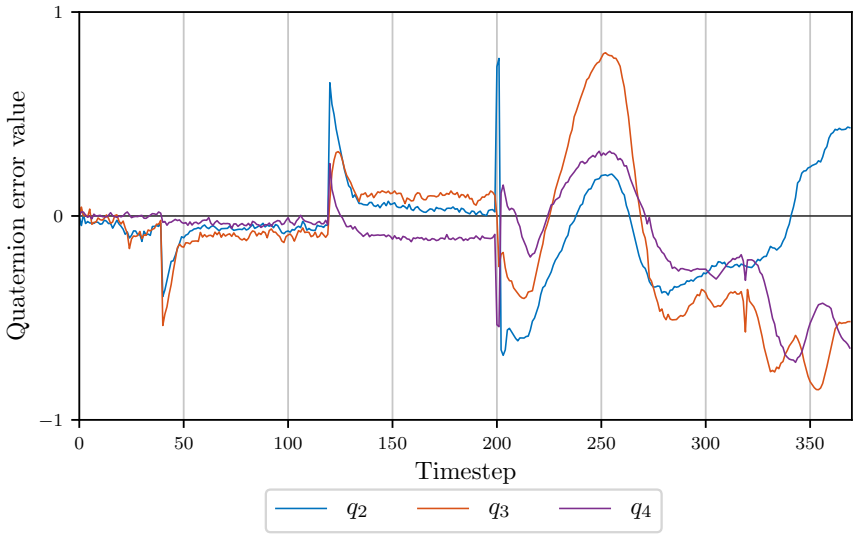


Figure 5.12 – Quaternion error between estimated and ground truth values due to the influence of outliers.

5.7 Conclusion

The EKF was introduced in this chapter along with the non-linear motion and measurement models. The EKF tracking algorithm was defined and used to solve the target localisation problem. The working of the algorithm and estimator was verified in a simulation environment. The simulation generated observation sets of the target, which were then provided to the estimator.

An initial distribution of the target pose is required and since this is relative to the sensor, the initial attitude was aligned with the CRF. The attitude quaternion was thus initialised with very small uncertainties, since the initial orientation is exact.

The mean Cartesian value of the first received measurements are used to initialise the distribution of the target position. Concatenating the measurements that are received for a few timesteps may also be used to perform a batch update of the initial position before tracking starts. The cameras will still only observe target planes facing towards them. This will result in a depth bias and the initial position that will always be closer than the true centre of rotation of the target, regardless of the number of features that were used to perform the update. More uncertainty is thus added to the initial depth of the target to allow the EKF to correct this over time. The estimator is tested in conjunction with the stereo-camera system using real world data in the next chapter.

Chapter 6

Experiments

6.1 Introduction

Results using simulated measurements were promising and in this chapter the efficacy of the system is tested using experimental data extracted from real world images. Using measurements from the stereo camera sensor may have more noise and imperfect correspondences. Illumination changes, shadows and glare are also present and can lead to further inaccuracies.

The experimental tests aim to use motions similar to the simulation. There are, however, a few limitations to the extent with which this can be executed. Recreating an ideal space environment without the effects of gravity and air friction is a laborious and expensive task. The experiment is set up to rotate the target with constant angular velocities, therefore the Newton-Euler dynamics that were tested in simulation, will not be investigated in this chapter. The experiment also tests displacements with constant velocity, unlike the sinusoidal translations that were tested in simulation.

A second constraint is the difficulty to obtain ground truth data of target motion. A possible solution is to reproject the estimated feature positions back onto the image plane. The reprojected points can then be compared to the observed features on the image plane and the error can be represented by the Euclidean distance between the predicted and actual points. This method can, however, not directly represent the tracked states or give an indication of the accuracies of the estimated values. Errors on the image plane can be caused by either rotation or translation estimation mistakes and there is no way to distinguish between this using the above-mentioned method. An alternative solution to this problem is presented in the next section.

6.2 Hardware Configuration

The robotic gantry, shown in Figure 6.1, was constructed to perform experimental tests. A calibrated stereo camera is configured to observe the target that is attached to the endpoint of the gantry. Two FLIR BlackFly cameras, shown in Figure 6.2, capture images of the target. The target is a small-scale satellite model with a shape similar to the simulated target depicted in Figure 5.3. A rotation stage is fixed to

a linear track and the translation is controlled by a stationary stepper motor. The rotation stage consists of a stepper motor and two angle adjustments, namely α and ψ. Using the angle adjustments shown in Figure 6.1 allows the use of one motor to generate three angular rates relative to the CRF.

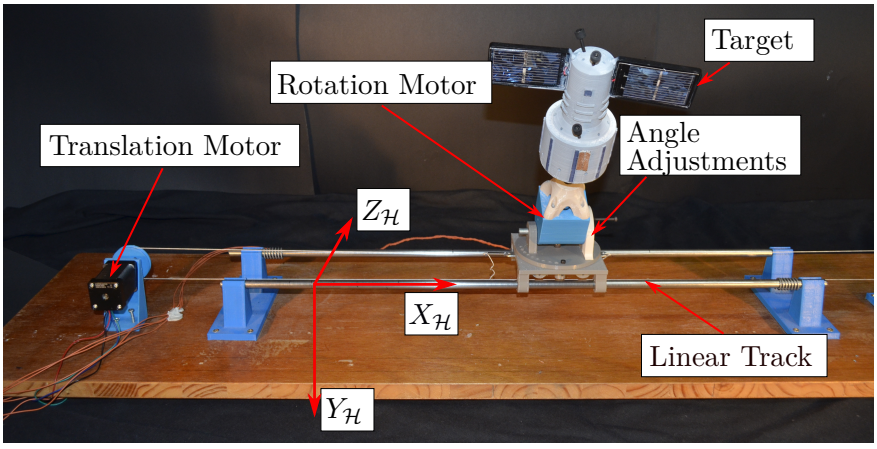


Figure 6.1 – Experimental hardware: Target fitted to the robotic gantry.

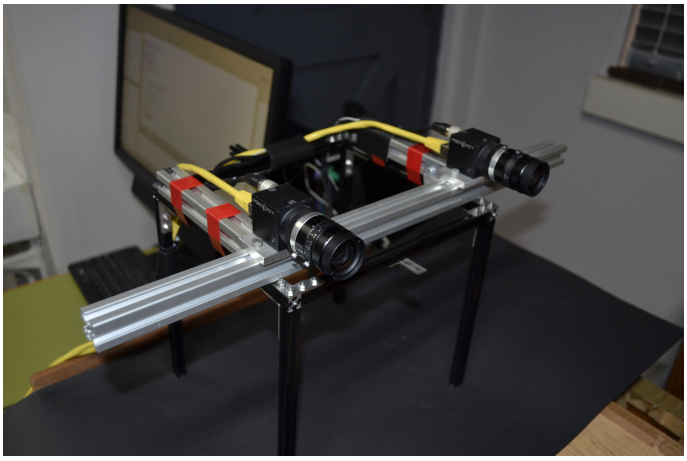


Figure 6.2 – Experimental hardware: FLIR BlackFly stereo camera pair.

A reference frame, $\mathcal{H}$, is fixed to the hardware and shown in Figure 6.1. The $\bar{\mathbf{x}}_{\mathcal{H}}$-axis is chosen to align with the translation direction of the linear track. The rotation between $\mathcal{H}$ and the camera is denoted by $\mathbf{A}_{\mathcal{C}}^{\mathcal{H}}$. An illustration of the hardware layout is shown in Figure 6.3. The cameras are fixed 100 mm apart in a stereo configuration. The robotic gantry is positioned with the linear rail roughly parallel to the $\bar{\mathbf{x}}_{\mathcal{C}}$-axis of the cameras. The alignment cannot be done exactly, which further motivates the need for a calibration step. The gantry is approximately 1500 mm away from the cameras. Additional light sources are placed next to the cameras to

illuminate the target in an otherwise dark room. The lights are covered with wax paper to diffuse the light that is projected onto the target.

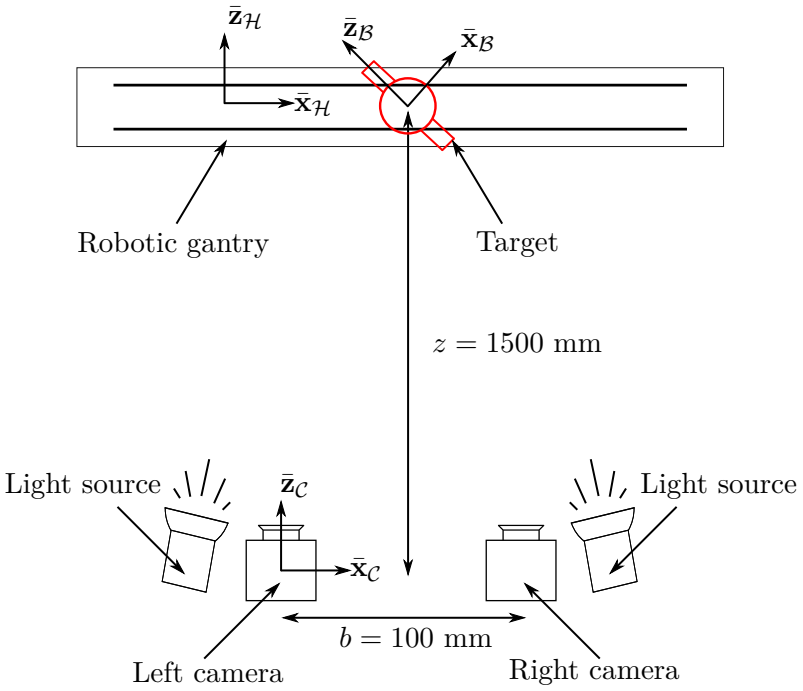


Figure 6.3 – Layout of the experimental hardware.

The rotation from $\mathcal{H}$ to the target, $\mathbf{A}_{\mathcal{H}}^{\mathcal{B}}$ is expressed using two coordinate rotations shown in Figure 6.4 as

$$\mathbf{A}_{\mathcal{H}}^{\mathcal{B}} = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & \sin(\psi) \\ 0 & -\sin(\psi) & \cos(\psi) \end{bmatrix} \tag{6.2.1}$$

$$= \begin{bmatrix} \cos(\alpha) & \sin(\psi)\sin(\alpha) & -\cos(\psi)\sin(\alpha) \\ 0 & \cos(\psi) & \sin(\psi) \\ \sin(\alpha) & -\sin(\psi)\cos(\alpha) & \cos(\psi)\cos(\alpha) \end{bmatrix}. \tag{6.2.2}$$

The motor rotates the target about the $\bar{\mathbf{y}}_{\mathcal{B}}$-axis with angular velocity ω_m. Using the coordinate transformations mentioned above, the angular velocity and attitude of the target in the CRF are described respectively as

$$\boldsymbol{\omega}_{\mathcal{B}/\mathcal{C}} = \mathbf{A}_{\mathcal{H}}^{\mathcal{B}} \boldsymbol{\omega}_{\mathcal{B}} = \mathbf{A}_{\mathcal{H}}^{\mathcal{B}} \begin{bmatrix} 0 \\ \omega_m \\ 0 \end{bmatrix} \quad \text{and} \tag{6.2.3}$$

$$\mathbf{A}_{\mathcal{C}}^{\mathcal{B}} = \mathbf{A}_{\mathcal{C}}^{\mathcal{H}} \mathbf{A}_{\mathcal{H}}^{\mathcal{B}}. \tag{6.2.4}$$

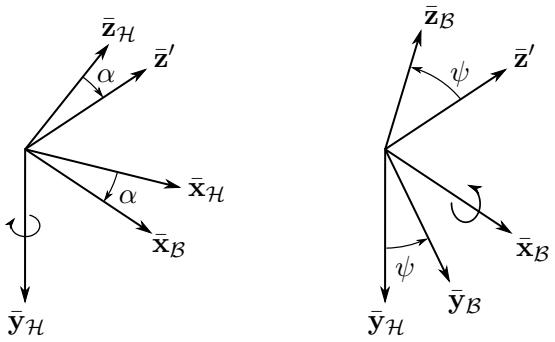


Figure 6.4 – Adjusting the axis of rotation using adjustment angles α and ψ.

The element values of $\mathbf{A}_{\mathcal{C}}^{\mathcal{H}}$ are obtained via calibration. A calibration board is fixed to the experiment hardware and aligned with the linear track to correspond to the hardware frame. Using a least-squares solver, the attitude of the board relative to the cameras is computed. Figure 6.5 shows the hardware axes projected onto the surface of the calibration board.

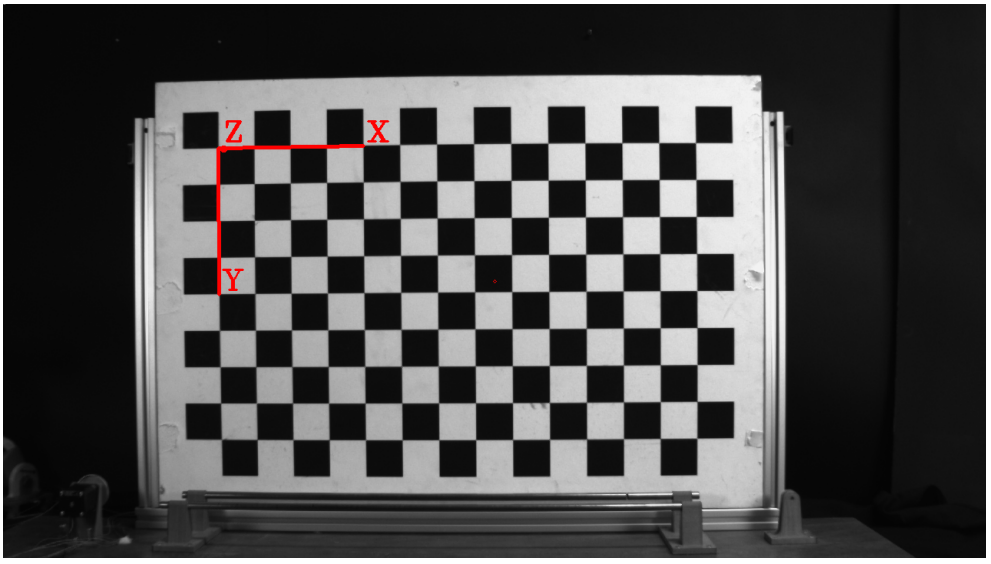


Figure 6.5 – Calibrating the hardware reference frame

System calibration errors may lead to offsets in the actual values of the rotation and translation of the target. Therefore, instead of only evaluating the estimated element values of the pose vector, the magnitudes of the angular and linear velocities are also evaluated. It is expected that these values should remain constant. This is a valid assumption since the target is moving at a constant velocity and therefore its momentum should remain constant.

6.3 Experimental Results

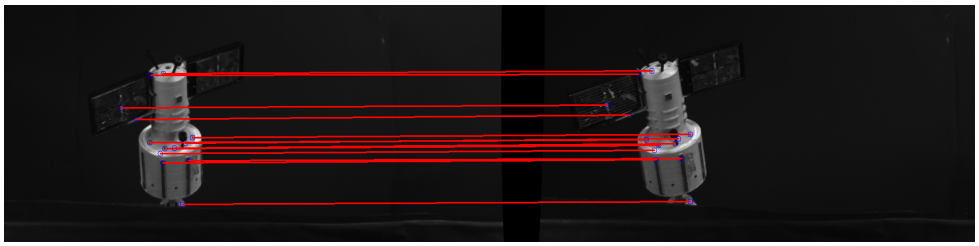
The experiment is performed over 200 timesteps, using the values shown in Table 6.1. The rotation motor has an angular velocity of 0.62 rad/s and the angular rates of the target in the CRF are calculated using the coordinate transformation in Equation 6.2.3. The target will complete one full rotation over the course of the experiment given its attitude rates. It is therefore expected that the attitude of the target at the start and end of the sequence will be the same, regardless of the orientation of the observer. The linear velocity of the target is parallel to the $\bar{\mathbf{x}}_{\mathcal{H}}$-axis. Due to practical constraints, the total displacement of the target over 200 timesteps is 240 mm, which is much less compared to the values used in simulation. Simulation has shown that large displacements can be successfully estimated.

$\boldsymbol{\omega}_{\mathcal{B}}$ (rad/s)	$\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}$ (rad/s)	$\mathbf{v}_{\mathcal{B}}$ (mm/s)	α	ψ	Timesteps	Step size (s)
$\omega_x = 0.0$	$\omega_1 = 0.003$	$v_x = 24$				
$\omega_y = 0.62$	$\omega_2 = 0.537$	$v_y = 0$	$0°$	$30°$	200	0.05
$\omega_z = 0.0$	$\omega_3 = 0.309$	$v_z = 0$				

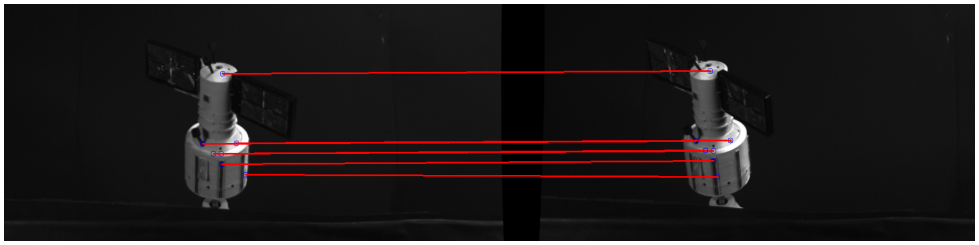
Table 6.1 – Experiment parameters

A sequence of stereo images captured by the cameras is shown in Figure 6.6. The images show the target translating from left to right and rotating around its $\bar{\mathbf{y}}_{\mathcal{B}}$-axis. It is evident that the rotation is not purely around only one axis relative to the cameras due to the angle adjustments. Stereo matches between the left- and right-hand images are indicated by red lines in the image sequence. The features in the left- and right-hand images have the same vertical positions, which confirms the implementation of the stereo rectification rule discussed in Section 4.2.

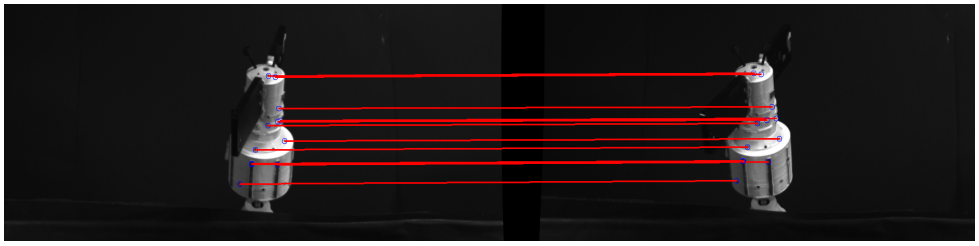
It is evident from the images that very few features are extracted on the solar panels. There are repetitive lines on the surfaces of the solar panels that do not yield strong corners for detecting features. The few features that are in fact detected are also often rejected by the brute-force matching algorithm. If the second best match for a feature is not significantly worse than the best match, then the match is rejected. This rejection frequently occurs when matches are made between non-unique features, like the tips of the lines on the solar panels. The solar panels are also very dark resulting in less features in those regions.



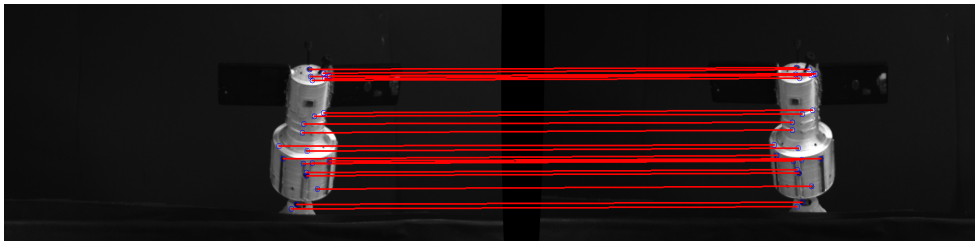
$t = 1$



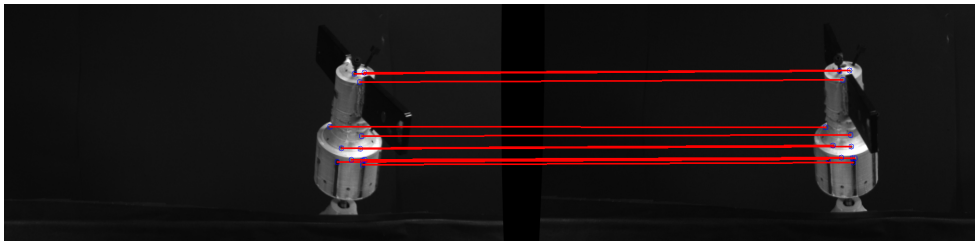
$t = 40$



$t = 80$



$t = 120$



$t = 160$

Figure 6.6 – Extracted stereo matches from experimental data sequence. Images are cropped for display purposes.

The feature extractor performs well and it is evident that the ratio of outliers to total matches is very small. The region of interest (ROI) algorithm, discussed in Section 4.3.4, is effective in isolating the moving target and the only features that

are extracted are located on the target. The feature extraction algorithm benefits from the featureless background.

This might not always be the case since the earth might be in the background when observing the target, depending on the relative orbit position between the target and the chaser satellite. Features that are detected in this case, may be eliminated by the following reasoning: If a feature is detected in both cameras and successfully matched, the depth of the feature may be calculated. Background features will have very large depth values, especially if they are detected on the surface of the earth. Any measurements with such high depth values may be discarded immediately by the stereo matching algorithm, since they will not lie within the range of the other measurements.

The number of extracted measurements for each timestep is plotted in Figure 6.7. Two valleys are created when the sides of the solar panels are facing toward the cameras, occluding a large part of the satellite body. There are always more than four measurements per timestep, except for a few single cases where it briefly reduces to three measurements. This is at the minimum points of the valleys when the solar panels cover most of the feature-rich regions of the satellite.

The number of features are within the same range as the occluded data from simulation. The number of measurements for the experimental case closely corresponds to Case 4 in Figure 5.10. Although the number fluctuates more than the simulated data, the average number of measurements is more than 12 for the duration of the experiment. It can be expected that tracking will succeed based on this information.

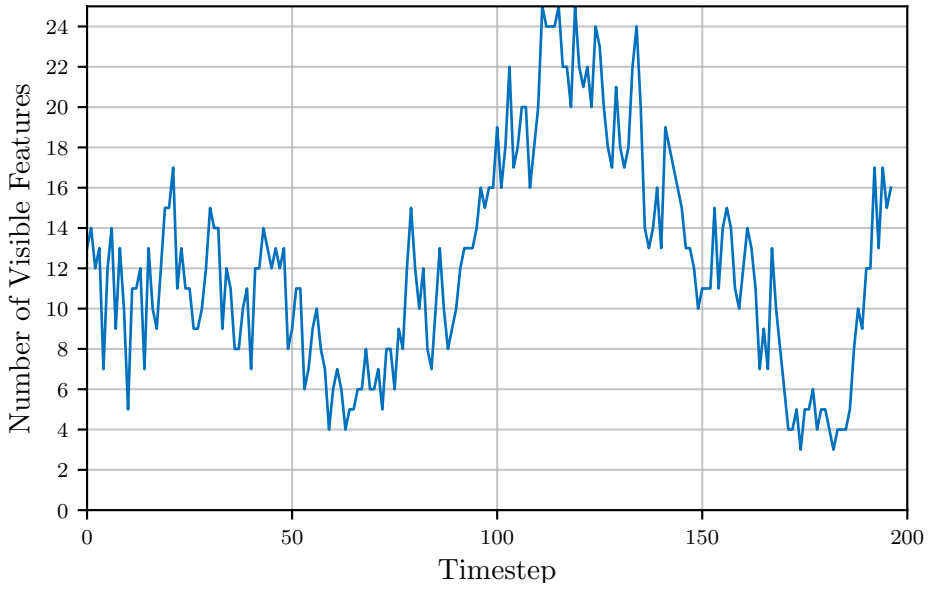


Figure 6.7 – Number of measurements per timestep, extracted from experimental data.

The estimated values of the quaternion attitude vector are shown in Figure 6.8. Estimated values are indicated with dotted lines. The plot shows acceptable tracking of the target attitude. Even though errors exist, the trend of the estimates are correct. The errors can be attributed to systemic errors in the calibration values of $\mathbf{A}_{\mathcal{C}}^{\mathcal{H}}$. Errors made in calibration will result in the wrong calculation of the expected angular rates, which in turn, are used to calculate the expected quaternion attitude values. It is clear from the figure that the estimated values meet up with the expected values at the end of the test at $t = 200$. This confirms that a full rotation was observed as expected when the experiment parameters were chosen.

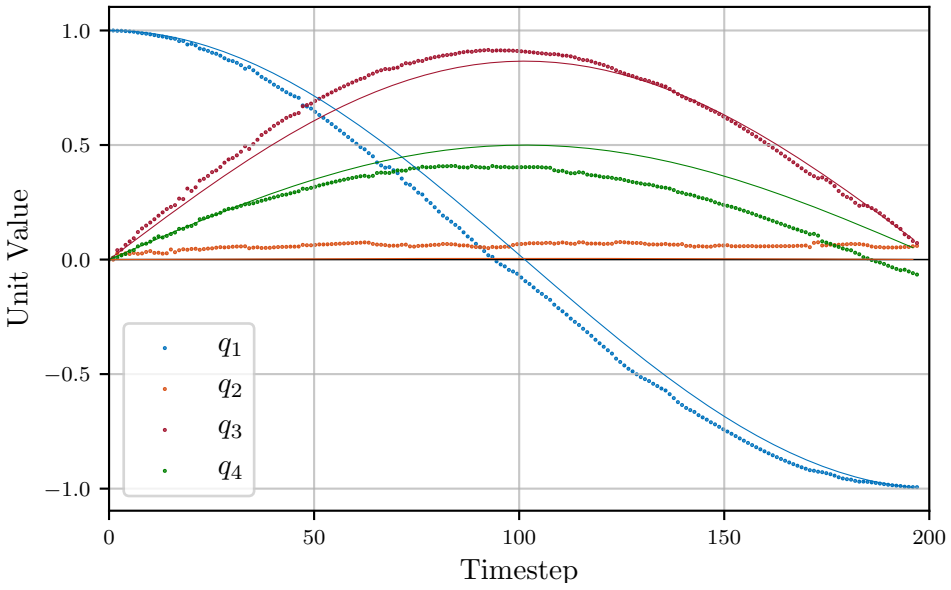


Figure 6.8 – Estimated quaternion attitude vector using real world data. The dotted lines indicate estimated values. Expected values are shown with solid lines in matching colour.

The magnitude of the angular rates $||\boldsymbol{\omega}_{\mathcal{C}}^{\mathcal{B}}||$, is displayed in Figure 6.9. The estimated magnitude is close to the velocity of the rotation motor, $\omega_m = 0.62$ rad/s. This result verifies that the estimator is tracking a constant angular velocity magnitude, as expected. The mean estimated values of the target's angular velocity vector are displayed in Figure 6.10. A clear difference when compared to the estimated angular velocities from simulation in Figure 5.5, is the absence of Newton-Euler coupling. The experiment hardware is not capable of stimulating Newton-Euler dynamics. Instead, the values of the angular rates are expected to remain constant, which is the case in results shown in Figure 6.10. The shaded regions in Figure 6.10 indicate the standard deviation over the mean values. The standard deviation decreases as the uncertainty of the angular rates become less.

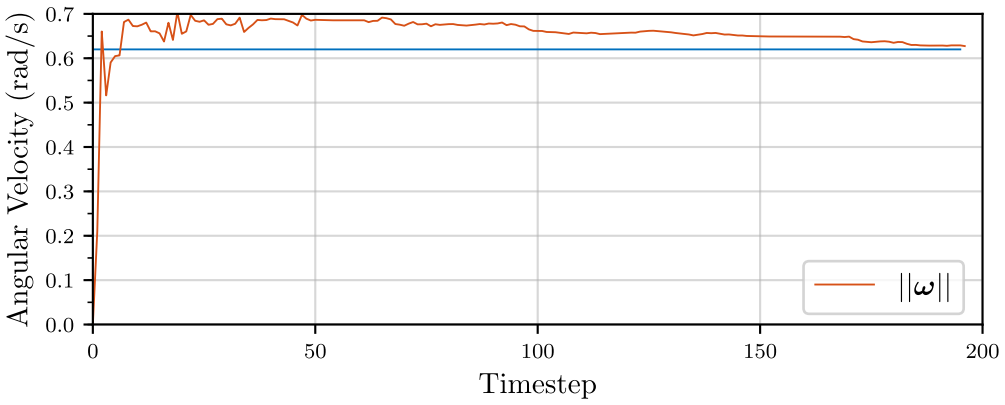


Figure 6.9 – Vector magnitude of the estimated angular velocities. The estimated values are indicated in orange and the expected angular rate magnitude is shown in blue.

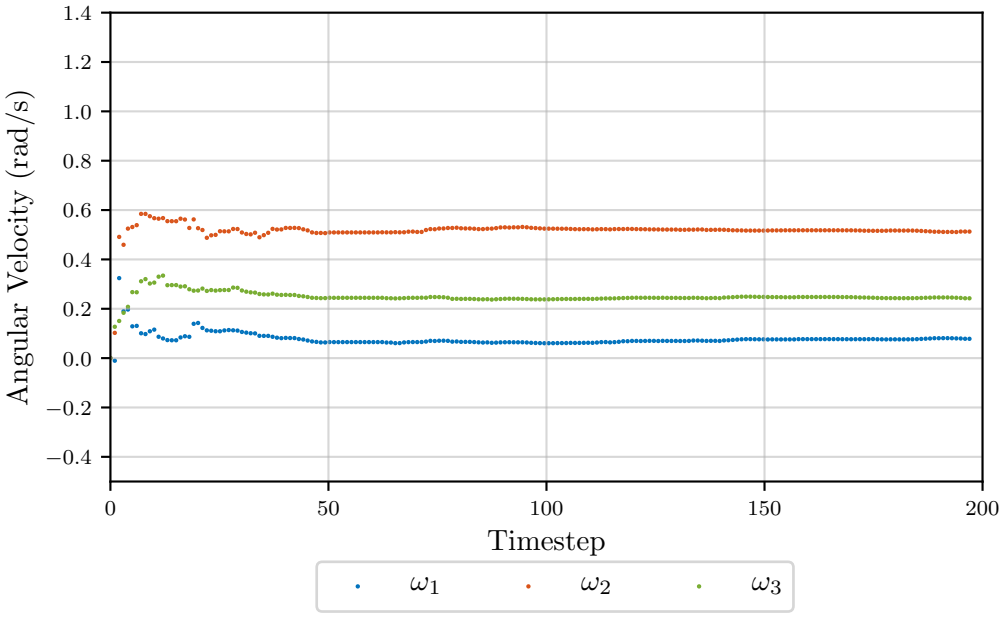


Figure 6.10 – Estimated angular rates using real world data. The shaded regions are used to indicate the standard deviation over the mean values of the estimated rates.

It is difficult to evaluate the quality of the absolute displacement estimates since the initial position of the target relative to the CRF is unknown. Instead, the relative displacement is evaluated because the exact distance that the target translates can be measured on the linear track. The nett Euclidean displacement between the

target's initial and final positions is calculated and directly compared to the expected translation in Figure 6.11.

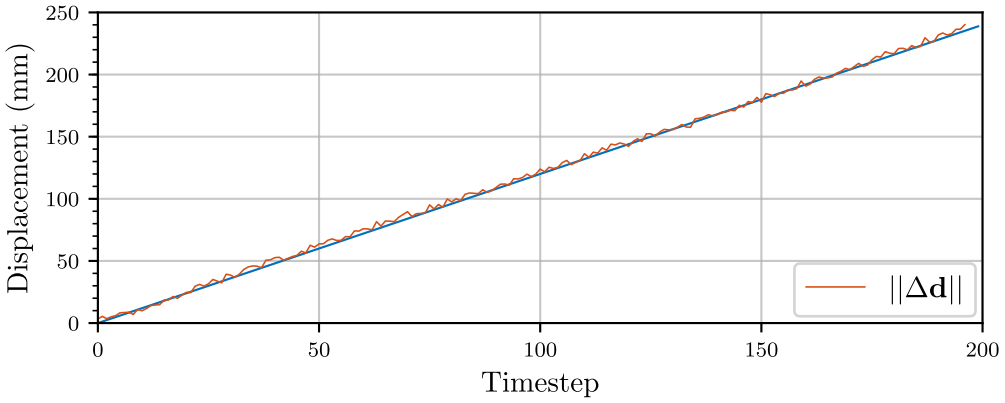


Figure 6.11 – Relative displacement magnitude between the target and the sensor. The orange line indicate the estimated nett displacement between the target and the sensor. The blue line indicates the expected displacement.

The estimated Euclidean displacement is very close to the real values. This result is expected since simulation results have shown that the estimator is capable of tracking complex translations with sinusoidal velocity profiles. Therefore, a linear translation estimate would not pose much of a challenge. To confirm the quality of the displacement estimates, the linear velocity magnitude is calculated and compared to the real value. The target moves with a linear velocity of 24 mm/s. The estimated magnitude converges to this value after approximately 40 steps, as shown in Figure 6.12.

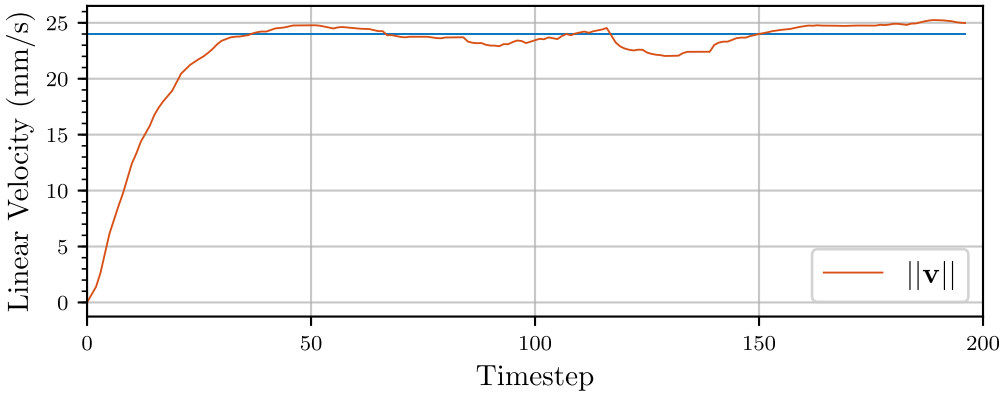


Figure 6.12 – Estimated linear velocity of the target using experimental data. The estimated value is expressed as a vector magnitude in orange. The expected linear velocity of the target is shown in blue.

6.4 Shape Reconstruction

Tracked target features are used with the estimated target pose information to create a body-centred reconstruction of the target's shape. To test the validity of the shape estimation, a rectangular cuboid is used as target to ease the interpretation of the 3D shape output. The target dimensions are 140 mm $\times$ 110 mm $\times$ 110 mm. Figure 6.13 provides several excerpts from the estimation sequence. Images from the left-hand camera are shown on the left, with a top and front view of the target data shown on the right. The front view shows the projected target features as seen down the bore-sight of the cameras. The stereo cameras are represented by the blue squares.

Figure 6.14 shows a perspective view of the reconstructed target. The true shape of the target is shown in red and the tracked features are indicated by the blue dots. A top down view of the reprojected features is shown in Figure 6.15. The quality of the reconstruction shows that the system is able to provide a partial shape estimate of the faces that were observed by the cameras. The difficulty of obtaining ground truth for pose estimation applies to the shape estimation verification as well. The orientation of the estimated shape and the true initial orientation of the target may have marginal offsets due to calibration errors.

The presence of outliers can be seen from the reconstructed top view. These outliers in the reconstructed shape may have been introduced to the system in the very last frames of the estimation sequence and the estimator might not have had enough time to remove them from the state vector or to improve the mean values of their position distributions.

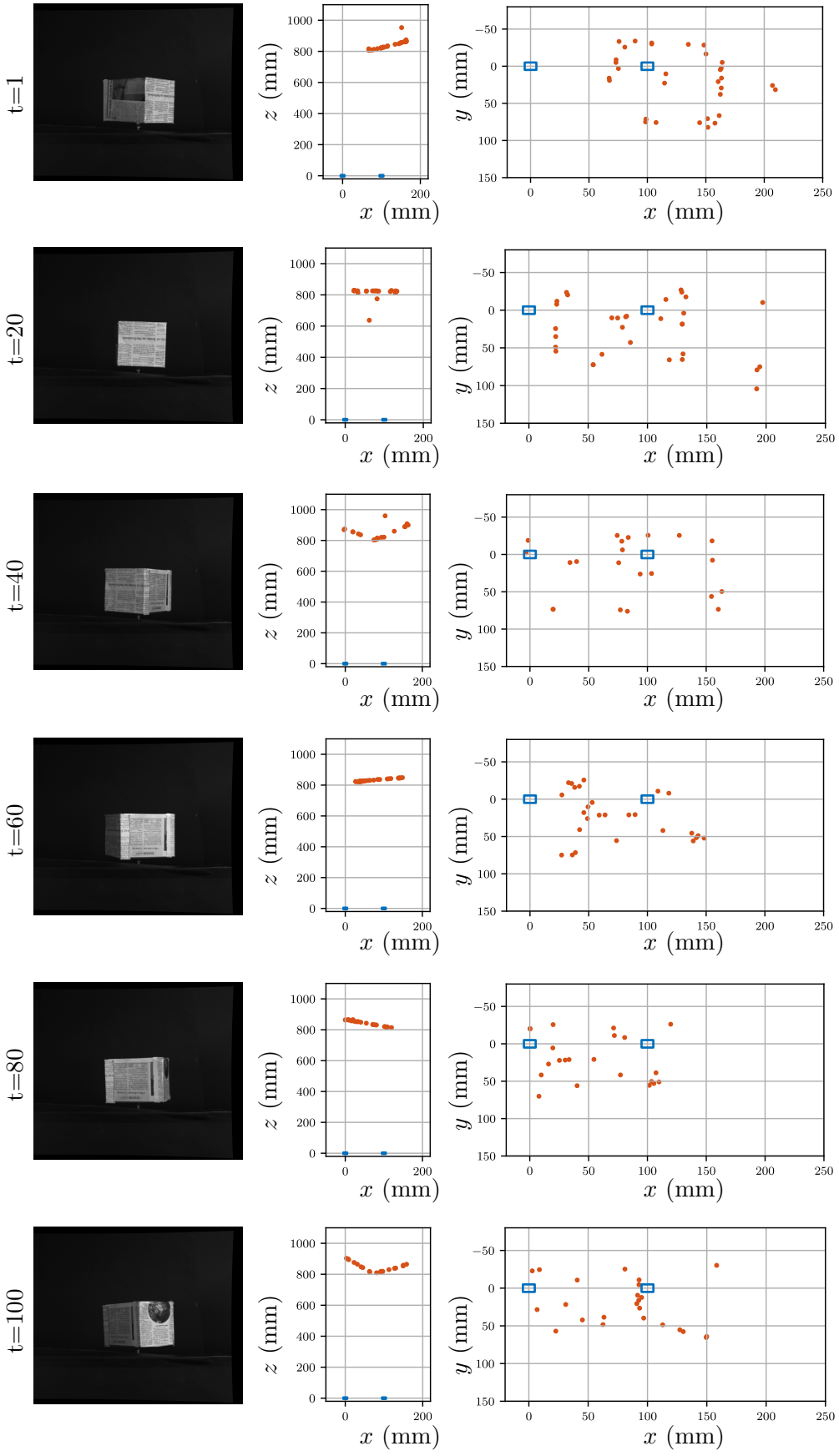


Figure 6.13 – Excerpts from target reconstruction sequence. Note that the images have been cropped for display purposes.

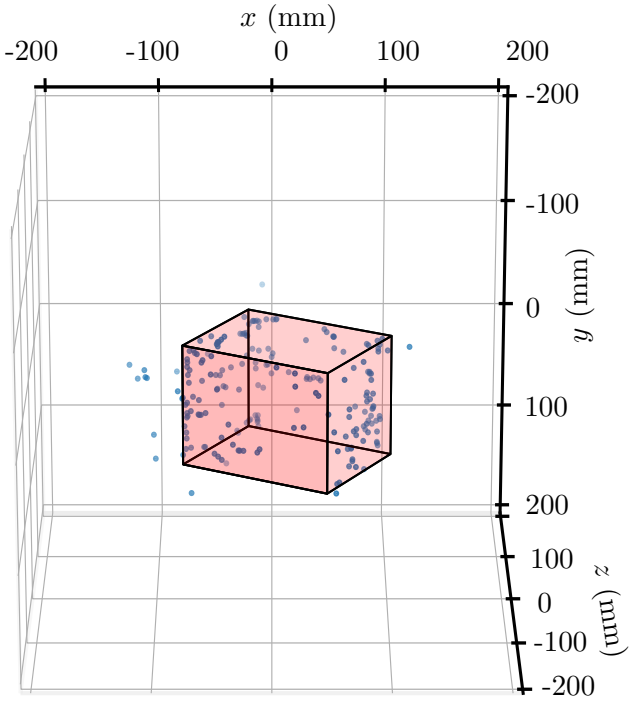


Figure 6.14 – Perspective view of reconstructed target. The expected shape is indicated by the red shaded regions. The estimated feature positions are shown in blue.

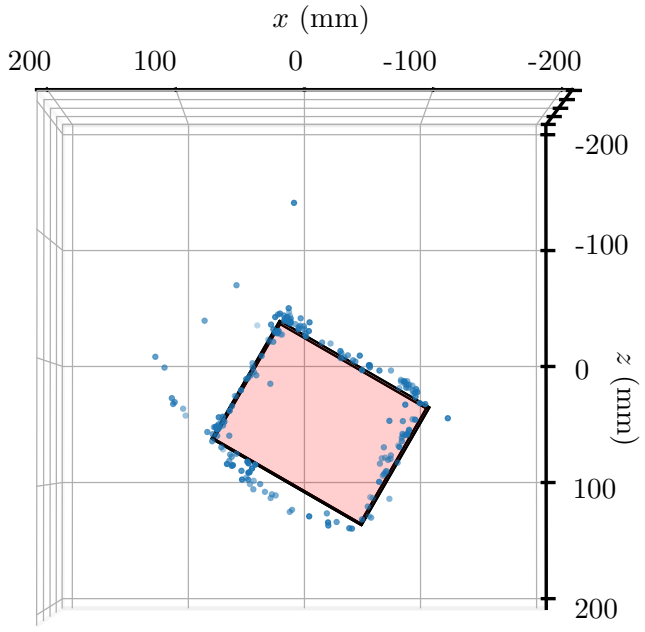


Figure 6.15 – Top view of reconstructed target. The expected shape is indicated by the red shaded regions. The estimated feature positions are shown in blue.

A root-mean-square (RMS) error may be calculated to assess the accuracy of the shape reconstruction, although it must be noted that a true representation of the accuracy is not possible without ground truth. The value is computed by determining the minimum distance of a feature to its ground truth plane on the target. Augenstein [45] proposes the normalisation of the RMS error using the distance to the target along the principal axis of the left camera ($z_{\mathcal{C}/\mathcal{B}}$). The error can, as a result, be directly compared to the accuracy that was computed from simulation.

The error for the experimental target reconstruction is calculated as:

$$\epsilon_{\text{exp}} = \frac{\epsilon_{\text{RMS}}}{z_{\mathcal{C}/\mathcal{B}}} = \frac{13.7 \text{ mm}}{931.3 \text{ mm}} = 0.01471. \tag{6.4.1}$$

The test is repeated in simulation using a target of the same shape and size as the experiment. The simulated target is also subjected to the same linear and angular velocities that was used in the experiment. The simulated target reconstruction yielded an RMS error of:

$$\epsilon_{\text{sim}} = \frac{\epsilon_{\text{RMS}}}{z_{\mathcal{C}/\mathcal{B}}} = \frac{7.4 \text{ mm}}{1000 \text{ mm}} = 0.0074. \tag{6.4.2}$$

It is difficult to make a direct comparison between the simulated and experimental results. The simulated data did not contain any outliers and an exact ground truth was available for comparison. Calibration errors in experimental data may have a large effect on the calculated reconstruction accuracy. Visual verification of the experimental reconstruction provides insight into the validity of the shape estimation.

6.5 Conclusion

This chapter described the experimental hardware used to verify the performance of the estimator when combined with the stereo-camera feature extractor. A method of evaluating the quality of the estimates without ground truth values was introduced. Experimental results showed that the system is able to track the target with acceptable accuracy.

Since the system makes use of imaging sensors and the data is processed one frame at a time, the accuracy of the system greatly depends on the framerate of the cameras. For this project, stepper motors were used to incrementally move the target and images were captured after each step. A realistically attainable framerate was chosen as 20 Hz. The experimental angular and linear velocities of the target purely depend on this chosen framerate. In practice, however, the exact framerate of the whole system, including the sensor and filter, would have to be accounted for and incorporated into a time-varying filter to accurately estimate the velocities. In the case where the target has very large angular velocities and the camera has a fairly low framerate, sampling errors may occur. The aliasing caused by observing a target with high velocities at a slow framerate, may lead to erroneous velocity estimates.

The system may benefit from the addition of an inertia estimator if enough information of the target has been accumulated as discussed in Section 5.5.1. This will only be a relative indication of the inertia tensor, since the mass of the target is still unknown. In the case where the inertia tensor becomes available, the estimator may change over to a dynamic estimator by using Newton-Euler dynamics in its motion model. There is, however, a risk when using an inertia that is computed solely from a sparse point cloud. Featureless regions on the target may not be tracked by the image processing algorithm, which can lead to a biased distribution of the true shape of the target. A summary of the work done in this project and concluding remarks are given in the next chapter.

Chapter 7

Conclusion and Future Work

This project developed a pose estimation system using vision sensors to track the states of a moving target for autonomous operations such as space debris removal, docking and satellite servicing. The shape, size and initial conditions of the target were unknown to the system prior to the start of the estimation procedure. Visual-based navigation is a well-researched topic in the field of robotics and an EKF-SLAM approach was implemented to perform estimation of a target's pose and shape relative to the sensor.

7.1 Conclusion

The impact of debris on further space exploration was investigated to assess the need for debris-related risk mitigation. It was found that the reduction of space debris that passive removal methods brings about is not sufficient to avoid the Kessler syndrome. Active debris removal (ADR) is a necessity to keep space accessible, especially in low earth orbits (LEO) where most debris is found. Two ADR missions, DEOS and RemoveDebris, were investigated to verify the need for accurate pose determination systems for ADR methods. Several of these ADR methods were assessed and it was found that inaccuracies in the respective pose estimation systems can lead to mission failure. Previously published research in the field of autonomous navigation and pose determination with space-based applications were investigated. Following this study the EKF was found to be a popular option when designing non-linear systems with spatial-based sensors such as cameras and Lidar.

The pose determination system that was developed in this project predominantly uses the EKF-SLAM algorithm. The measurement model of the classic autonomous navigation SLAM problem was inverted to a solution where the landmarks/features move relative to the stationary robot/sensor. The movement of the target features are used to estimate not only the positions of the features relative to the target reference frame, but also the motion of the target relative to the CRF. The non-linear propagation of the target's angular position and velocities along with a non-linear measurement model necessitates the use of linearisation techniques to update the state covariance. The Taylor series approximation was used in the EKF to accomplish this.

A kinematic motion model was used since no prior information of the target was available. The relative moment of inertia information was not available and therefore the angular velocities are updated using a linear model, instead of the faster converging dynamic model. The dynamic motion model leads to more accurate tracking of nutations since it better describes the Newton-Euler coupling between the angular rates.

The pinhole camera model and stereo geometry were introduced and used the image plane projections of the target's features to calculate 3D positions of the features. A robust feature extraction framework was developed that not only successfully performs temporal extraction of SIFT features, but also performs matching across left and right camera frames. This provides measurements to the estimation system in the form of 3D target feature positions with correspondence. Establishing reliable measurement correspondence is vital for the successful operation of the estimation algorithm.

The full state non-linear motion and measurement models were implemented in the EKF-SLAM algorithm. Estimation was done using both simulated and experimental data. Torque-free motion in a gravity- and friction-free environment was tested in simulation. Performance tests were also done in simulation to determine the minimum number of measurements that are required to perform accurate estimation. Simulation also determined the extent to which the system is invariant to outliers due to noisy measurements.

Finally, a practical experiment was created that tested the integration of the feature extraction algorithm with the pose estimation system. Calibration was done to obtain an indication of the expected mean values of the state vector. Results showed that the system was capable of estimating the relative orientation and position of the target relative to the stereo camera sensors with the same magnitudes and trends as the expected values. RMS errors of sparse shape and size reconstruction of the target showed a close correlation between the simulated and experimental data.

A sensible indication over the initial distribution of the target's states is required. The mean of the target attitude was initialised with the same orientation as the CRF. This relative attitude is exact and therefore very small uncertainties are introduced over the quaternion attitude. The relative displacement was initialised with a mean equal to the average distance to the target. This average distance was calculated using the mean position of the first set of measurements. Features are only detected on planes facing the cameras, resulting in a bias in the initial depth. The initial depth of the target's centre of rotation is therefore initialised with a larger uncertainty than the horizontal and vertical positions.

7.2 Future Work

Data of the moving target is obtained using stop motion images with the stepper motors and the stereo-camera, as mentioned in Section 6.2. All processing is performed offline after the data capturing process is completed. Image processing was done in C++ and the estimation algorithm was implemented in Python. Integrating the two separate systems and executing it as a single system in a faster language

will be required to perform on-line estimation. The addition of real-time estimation will enable the execution of more interesting experiments, such as the capturing of a target with a robot on a frictionless air bed. More hardware development is also required to test the system with better expected truth values.

The system will benefit greatly from an inertia estimator, especially when observing coupled dynamics. Maximum likelihood estimation may be used to identify an inertia tensor that best relates to the observed motion. Occasionally resetting the target reference frame may help to reduce drifting of the quaternion attitude. The distribution over the angular rates can be retained, which will in turn help to better estimate the quaternion attitude.

A probabilistic representation of the target structure can be created to describe the features in the body-fixed reference frame. This can be done if the target is observed for prolonged periods of time. This point cloud can be used to calculate and update the relative moments of inertia. The point cloud can alternately be used for point cloud matching algorithms like the iterative closest point (ICP) algorithm or point set registrations. These algorithm can provide a filtering algorithm with coarse pose estimates, which can then be refined to extract more accurate estimates.

Bibliography

[1] Lafleur, C.: A Comprehensive Census of All Spacecraft Ever Launched. The Spacecraft Encyclopedia, 2017. Accessed on 2018-07-25.
Available at: `http://claudelafleur.qc.ca/Spacecrafts-index.html`

[2] SpaceX: Falcon 9. From SpaceX website, 2017. Accessed on 2018-11-14.
Available at: `https://www.spacex.com/falcon9`

[3] Garcia, M.: Space Debris and Human Spacecraft. 2017. Accessed on 2018-07-26.
Available at: `https://www.nasa.gov/mission_pages/station/news/orbital_debris.html`

[4] Opromolla, R., Fasano, G., Rufino, G. and Grassi, M.: A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Progress in Aerospace Sciences*, vol. 93, no. February, pp. 53–72, 2017.

[5] Mokuno, M. and Kawano, I.: In-orbit demonstration of an optical navigation system for autonomous rendezvous docking. *Journal of Spacecraft and Rockets*, vol. 48, pp. 1046–1054, 2011.

[6] Reintsema, D., Sommer, B., Wolf, T., Theater, J., Radthke, A., Sommer, J., Naumann, W. and Rank, P.: DEOS - The In-Flight Technology Demonstration of German's Robotics Approach to Dispose Malfuntioned Satellites. *11th Symposium on Advanced Space Technologies in Robotics and Automation*, 2011.

[7] Forshaw, J., Aglietti, G., Salmon, T., Retat, I., Chabot, T., Pisseloup, A., Phipps, A., Bernal, C., Chaumette, F., Forshaw, J., Aglietti, G., Salmon, T., Retat, I. and Burgess, C.: The RemoveDebris ADR Mission : Launch from the ISS , Operations and Experimental Timelines. *IAC 2018 - 68th International Astronautical Congress*, pp. 1–9, Sep 2017.

[8] Kessler, D.J. and Cour-Palais, B.G.: Collision Frequency of Artificial Satellites: the Creation of a Debris Belt. *Journal of Geophysical Research*, vol. 83, no. A6, pp. 2637–2646, 1978.

[9] Weeden, B.: Anti-Satellite Tests in Space - The Case of China. *Secure World Foundation*, 2013.

Available at: `https://swfound.org/media/115643/china_asat_testing_fact_sheet_aug_2013.pdf`

[10] Kelso, T.S.: Analysis of the Iridium 33-Cosmos 2251 collision. *Advances in the Astronautical Sciences*, vol. 135, pp. 1099–1112, 2010.

[11] Anz-Meador, P.: Orbital Debris Quarterly News. *National Aeronautics and Space Administration*, vol. 22, pp. 10–11, 2018.

[12] Liou, J.-C.: Orbital Debris Quarterly News. *National Aeronautics and Space Administration*, vol. 17, no. March, pp. 1–10, 2012.

[13] Forshaw, J.L., Aglietti, G.S., Navarathinam, N., Kadhem, H., Salmon, T., Pisseloup, A., Joffre, E., Chabot, T., Retat, I., Axthelm, R., Barraclough, S., Ratcliffe, A., Bernal, C., Chaumette, F., Pollini, A. and Steyn, W.H.: RemoveDEBRIS: An in-orbit active debris removal demonstration mission. *Acta Astronautica*, vol. 127, pp. 448–463, 2016.

[14] Taylor, B., Aglietti, G., Fellowes, S., Salmon, T., Hall, A., Chabot, T., Pisseloup, A., Ainley, S., Tye, D., Retat, I., Bernal, C., Chaumette, F., Pollini, A. and Steyn, W.H.: Removedebris preliminary mission results. *69th International Astronautical Congress*, pp. 1–5, October 2018.

[15] Shan, M., Guo, J. and Gill, E.: Review and comparison of active space debris capturing and removal methods. *Progress in Aerospace Sciences*, vol. 80, pp. 18–32, 2016.

[16] Hakima, H. and Emami, M.R.: Assessment of active methods for removal of LEO debris. *Acta Astronautica*, vol. 144, no. December 2017, pp. 225–243, 2018.

[17] Malan, D.F.: *3D Tracking between Satellites using Monocular Computer Vision*. Master's thesis, University of Stellenbosch, Dept. Elect. Eng, 2004.

[18] Song, J. and Cao, C.: Pose Self-Measurement of Noncooperative Spacecraft Based on Solar Panel Triangle Structure. *Journal of Robotics*, 2015.

[19] Oumer, N.W., Kriegel, S., Ali, H. and Reinartz, P.: Appearance learning for 3D pose detection of a satellite at close-range. *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 125, pp. 1–15, 2017.

[20] Dong, G. and Zhu, Z.H.: Autonomous robotic capture of non-cooperative target by adaptive extended Kalman filter based visual servo. *Acta Astronautica*, vol. 122, pp. 209–218, 2016.

[21] Lichter, M. and Dubowsky, S.: State, shape, and parameter estimation of space objects from range images. *IEEE International Conference on Robotics and Automation*, pp. 2974–2979 Vol.3, April, 2004.

[22] He, Y., Liang, B., He, J. and Li, S.: Non-cooperative spacecraft pose tracking based on point cloud feature. *Acta Astronautica*, vol. 139, pp. 213–221, 2017.

[23] Shahid, K. and Okouneva, G.: Intelligent LIDAR scanning region selection for satellite pose estimation. *Computer Vision and Image Understanding*, vol. 107, no. 3, pp. 203–209, 2007.

[24] Woods, J.O. and Christian, J.A.: Lidar-based relative navigation with respect to non-cooperative objects. *Acta Astronautica*, vol. 126, pp. 298–311, 2016.

[25] Pesce, V., Lavagna, M. and Bevilacqua, R.: Stereovision-based pose and inertia estimation of unknown and uncooperative space objects. *Advances in Space Research*, vol. 59, no. 1, pp. 236–251, 2017.

[26] Biondi, G., Mauro, S., Mohtar, T., Pastorelli, S. and Sorli, M.: Feature-based estimation of space debris angular rate via compressed sensing and Kalman filtering. *3rd IEEE International Workshop on Metrology for Aerospace, MetroAeroSpace 2016 - Proceedings*, pp. 215–220, 2016.

[27] Segal, S., Carmi, A. and Gurfil, P.: Stereovision-based estimation of relative dynamics between noncooperative satellites: Theory and experiments. *IEEE Transactions on Control Systems Technology*, vol. 22, no. 2, pp. 568–584, 2014.

[28] Chen, S.Y.: Kalman filter for robot vision: A survey. *IEEE Transactions on Industrial Electronics*, vol. 59, no. 11, pp. 4409–4420, 2012.

[29] Brink, W.: *Stereo vision for simultaneous localization and mapping.* Master's thesis, University of Stellenbosch, Dept. Elect. Eng, 2012.

[30] Thrun, S., Burgard, W. and Fox, D.: Probailistic Robotics. 2000.

[31] Zeng, A., Song, S., Yu, K., Donlon, E., Hogan, F.R., Bauzá, M., Ma, D., Taylor, O., Liu, M., Romo, E., Fazeli, N., Alet, F., Dafle, N.C., Holladay, R., Morona, I., Nair, P.Q., Green, D., Taylor, I., Liu, W., Funkhouser, T.A. and Rodriguez, A.: Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *CoRR*, 2017. Available at: `https://arxiv.org/abs/1710.01330`

[32] Katz, D., Kazemi, M., Bagnell, J.A. and Stentz, A.: Clearing a pile of unknown objects using interactive perception. *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 154–161, 2013.

[33] Jiang, W., Ferrero, A., Salicone, S. and Zhang, Q.: A Possible Way to Perform Recursive Bayesian Estimate in the Possibility Domain. *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 12, pp. 3218–3227, 2017.

[34] Van Daalen, C., Jones, T. and Jaquet, C.: Advanced Estimation 813 : Course Notes. University of Stellenbosch, Dept. Elect. Eng. 2017.

[35] Diebel, J.: Representing attitude: Euler angles, unit quaternions, and rotation vectors. Stanford University. October 2006.

[36] Sidi, M.J.: *Spacecraft Dynamics & Control.* Cambridge, 1997.

[37] Marsden, J.E. and Ratiu, T.S.: Introduction to Mechanics and Symmetry. *Physics Today*, vol. 48, no. July, p. 70, 1998.

[38] Zhang, Z.: A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[39] Zucchelli, M.: *Optical Flow Based Structure From Motion.* Ph.D. thesis, Royal Institute of Technology, Stockholm, 2002.

[40] Fleet, D. and Weiss, Y.: *Optical Flow Estimation*, pp. 237–257. Springer US, Boston, MA, 2006.

[41] Patel, D. and Saurahb, U.: Optical flow measurement using Lucas Kanade method. *International Journal of Computer Applications*, vol. 61, no. 10, pp. 6–10, January 2013.

[42] Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L.: Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[43] Lowe, D.G.: Distinctive image features from scale invariant keypoints. *International Journal of Computer Vision*, vol. 60, pp. 91–11020042, 2004. 0112017.

[44] Leutenegger, S., Chli, M. and Siegwart, R.Y.: CV Reading BRISK : Binary Robust Invariant Scalable Keypoints. *Proceedings of the IEEE International Conference on Computer Vision.*, pp. 2548–2555, 2011.

[45] Augenstein, S.: *Monocular pose and shape estimation of moving targets, for autonomous rendezvous and docking.* Ph.D. thesis, Stanford University, 2011.