Relative Navigation for Debris Removal (RENDER)

Designing a Novel Fiducial Marker System

David Rijlaarsdam

Technische Universiteit Delft

TU Delft

Cover Image: Artist rendition of Mantis Marker on Envisat. While the marker depicted is quite large with respect to the spacecraft for artistic purposes, in reality it would be much smaller and distributed. Background taken from NASA AS17-148-22688 (7-19 Dec. 1972). The white dots surrounding the spacecraft are debris from the Saturn S-IVB stage separation.

Relative Navigation for Debris Removal (RENDER)

Designing a Novel Fiducial Marker System

by

David Rijlaarsdam

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday July 9, 2021 at 10:00 AM.

Student number:	4307127	
Project duration:	September 1, 2020 – June 25, 2021	
Thesis committee:	Prof. dr. E.K.A. Gill,	TU Delft, chair
	Dr. ir. J.M. Kuiper,	TU Delft, supervisor
	M. Zwick, M. Eng.	European Space Agency, supervisor
	Dr. ir. W. van der Wal,	TU Delft, examiner

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft esa

Abstract

To mitigate the growth of space debris, it is vital that Active Debris Removal (ADR) is performed in the near future. Since ADR will require robust relative navigation, future spacecraft should be marked with fiducial markers that are robustly extracted from an image taken by a chaser spacecraft. To prevent the restriction of robotic movement and occlusions of features, planarity of these fiducials is desired. While numerous planar fiducials exist, the performance of these systems suffers from pose ambiguities and loss of precision under frontal observations. In order to mitigate these issues, encoding markers have been proposed in literature. These markers encode an extra dimension of information in the signal between marker and sensor, thus increasing the robustness of the fiducial system. However, little research has been done on these encoding markers and existing solutions are hard to manufacture, qualify and scale.

This thesis proposes a novel fiducial marker design based on the compound eye of a Mantis insect. The fiducial proposed in this work, the so-called Mantis Marker, is planar and encodes a virtual point in its signal. This virtual point is located at a distance from the marker plane. This makes the marker system mathematically equivalent to a fiducial that uses a protrusion or indentation. This increases pose robustness by an order of magnitude with respect to planar fiducials of the same dimensions, as demonstrated in this thesis. Furthermore, the marker is easy to manufacture, is scalable and has low complexity. Finally, the marker is suitable for pose estimation for robotics and Augmented Reality (AR) on Earth where additional pose robustness is required.

Preface

To conclude my student life with a research period at the European Space Agency has been a great privilege. This thesis and the work on the *Mantis Marker* would have never come to fruition without the many amazing people around me, both professionally and personally. I would like to highlight a few of the people that have helped me write this thesis.

First of all, I would like to thank my supervisors: Martin Zwick and Hans Kuiper. They have given me enough freedom to steer my research while guiding me in the right direction wherever necessary. I know for a fact that not all of my peers have been as lucky with their supervision during their thesis, and I am very grateful for it.

I would like to thank Marti Vilella for the fun times and for showing me around ESTEC. The lab work would not have happened without you, nor would it have been as much fun.

In no particular order, I would like to thank all the people that have helped me with their tremendous expertise throughout my thesis: Andrew Wolahan, Paul Guitrand, Manuel Sanchez Gestido, Tobias Boenke, Lorenzo Pasqualetto, Martin Schwendener, Ana Cipriano and Irene Huertas.

TEC-MMA has been an amazing team to be a part of, even temporarily. I want to thank everyone from the section for making my time both working remotely and on-site as fun as it was. I hope to be back one day.

Last but not least, I want to thank my family and friends for being there for me.

To paraphrase Buzz Aldrin:

"I hope that this work may in some way contribute to the exploration of space. If only I could join these exciting endeavours!"

David Rijlaarsdam
Rotterdam, June 2021

Contents

Acronyms

ADR Active Debris Removal. iii, 1–5, 7–9, 14, 15, 17–21, 24, 26, 30, 37, 91, 93

AHP Analytic Hierarchy Process. 4, 31, 33–39

AR Augmented Reality. iii

ATV Automated Transfer Vehicle. 8

AVGS Advanced Video Guidance Sensor. 32

BRDF Bidirectional Reflectance Distribution Function. 60

CNN Convolutional Neural Network. 10

COTS Commercial of the Shelve. 26, 91

D4R Design for Removal. 7, 14

DLS Direct Least-Squares. 12, 13

DOF Degree Of Freedom. 9, 11, 21, 24, 48

DOT Design Option Tree. 24, 30, 31

ECSS European Cooperation for Space Standardization. 6

EO Earth Observation. 1

ESA European Space Agency. 19, 65

ESTEC European Space Research and Technology Centre. 65

FOV Field Of View. 13, 22, 28, 48, 49, 54, 55, 57, 59, 66, 71, 83, 84, 91, 92

GEO Geostationary Earth Orbit. 6

GNC Guidance, Navigation and Control. 20, 66

GNSS Global Navigation Satellite System. 1

GPS Global Positioning System. 8, 15

GRALS GNC Rendezvous, Approach and Landing Simulator. 66

IMU Inertial Measurement Unit. 8, 22

INCOSE International Council on Systems Engineering. 17

ISS International Space Station. 8, 32

KPPs Key Performance Parameters. 26, 33, 37, 39

LED Light Emitting Diode. 8, 59, 62, 63

LEO Low Earth Orbit. 5–9, 14, 21, 24, 30, 91

LIDAR Light Detection And Ranging. 10

LiS Lost in Space. 22, 30

MEO Medium Earth Orbit. 6

MLI Multi Layer Insulation. 59

MVP Minimum Viable Product. 44, 45

OOS On-Orbit Servicing. 1, 8

ORGL Orbital Robotics and GNC Lab. 65, 71

PnP Perspective-N-Point. 3, 5, 13, 15, 26, 28, 48, 58, 75, 79, 84

POS Pose from Orthography and Scaling. 12

RENDER Relative Navigation for Debris Removal. 4, 17–22, 26, 30–33, 39, 45, 46, 65, 66, 70

RF Radio Frequency. 8, 15

SVMS Space Vision Marker System. 14, 26, 48

SVS Space Vision System. 32

TRL Technology Readiness Level. 33, 37

UNOOSA United Nations Office for Outer Space Affairs. 5

VBS Visual Based Sensor. 24, 59

Nomenclature

$\delta_{k,i,j}$	Minimum change in weight factor k that causes rank reversal between option i and j
λ	Wavelength in nm
λ_{max}	Principal eigenvalue comparison matrix AHP
Ω_i	Solid angle of i in sterradians
ψ	Yaw in degrees
θ	Pitch in degrees
A_d	Detector area
a_h	Horizontal slope of Aruco side in image frame
a_{ij}	Measure of performance of design option i for criterion j for AHP
A_{marker}	Marker area in m^2
a_v	Vertical slope of Aruco side in image frame
c	Speed of light
c_j	Scalar value of Aruco corner coordinates j
c_x, c_y	Coordinates of the principal point on the image plane
CI	Consistency Index AHP
CR	Consistency Ratio AHP
Dd_h	Horizontal dimension of detector
Dd_v	Vertical dimension of detector
E	Irradiance in W/m^2
f_x, f_y	Focal length in pixels
g	Gap between Aruco and encoding element divided by Aruco width
h	Height of encoding element divided by Aruco width
h_{Planck}	Planck constant
I	Camera intrinsic matrix
$I_{radiant}$	Radiant Intensity in W/sr
L_i	Radiance of i in $W/m^2/sr$
m_k^i	Scalar value of coordinates of encoding element corner k for element i
n_{max}	Gaussian noise standard deviation on feature point in pixels at maximal range
n_{min}	Gaussian noise standard deviation on feature point in pixels at minimal range

Ps_x	Horizontal pixel size
Ps_y	Vertical pixel size
Psi	Flux density in photons per second
Qe	Quantum efficiency detector
R_{ct}	Rotation matrix chaser w.r.t. target
r_{ct}	Translation vector chaser w.r.t. target
R_i	Range i of relative pose estimation
Res_h	Horizontal resolution of detector
Res_v	Vertical resolution of detector
RI	Random Index AHP
s	Scale factor
T_p	Throughput
T_{ct}	Homogeneous transformation matrix chaser w.r.t. target
u, v	Coordinates in the image frame
u_i	x coordinate of ith point in image frame
u_k^v	x coordinate of kth virtual point
V_d	Depth of the virtual point
v_i	y coordinate of ith point in image frame
v_k^v	y coordinate of kth virtual point
w_{bit}	Width of a bit of the encoding element of the marker in metres

1 Introduction

Humankind is becoming more dependant on access to space. Apart from scientific value, exploration opportunities and political prestige, space is a vital asset with real economic value. This value comes from multiple factors: position determination using Global Navigation Satellite System (GNSS) is vital for nearly every industry and consumer, policymakers rely on spacecraft to give insight into ecological processes, communication and internet access is provided through constellations and insurance companies and militaries rely on high resolution Earth Observation (EO) data.

However, while the value of space is undeniable, the increase of exploitation of this resource is not without costs. The orbital space around Earth is finite: with the increase of the number of objects around Earth (spacecraft, rocket stages, debris etc.) the chances of collision between objects increases as well. This presents not only a clear and present danger to existing and future spacecraft, but a more fundamental problem: losing access to space from Earth.

1.1. Motivation

The issue of space debris can no longer be mitigated with current policies. Even if all launches are halted, the risk of collision will increase for the coming 200 years [1]. Clearly, halting all launches is not realistic: the number of new objects in space is increasing every year. Therefore, additional measures are necessary.

The most effective method to reduce the number of objects in space is debris mitigation measures. These measures are policies for spacecraft that ensure that a minimum amount of new objects is released per mission. However, these measures are not feasible for all new objects (such as rocket adapters) nor are they fully reliable. A spacecraft may fail before end-of-life. In some cases, such a failure can be mitigated by On-Orbit Servicing (OOS), but not in every instance. This shows the need for Active Debris Removal (ADR): to ensure future access to space, it is vital that existing and future objects are removed from orbit.

A number of methods for ADR have been explored [2]. Arguably the most feasible option is the removal of objects by capturing these objects using a purposely designed removal spacecraft. Such a spacecraft requires robust relative navigation to rendezvous with a target object.

In order to facilitate robust relative navigation for both OOS and ADR, the target object should be marked with features that are robustly extracted from an image taken by a chaser spacecraft. Numerous of these marking features, or fiducials, exist for robotic systems both for Earth applications and in spaceflight. An example is shown in Figure 1.1. In many applications, these fiducials need to be planar. Planarity enables a reduction of marker volume and reduces the chances of pose estimation failure due to occlusion as well as ensuring that

robotic movement is not blocked.

Existing passive fiducial systems such as ARtoolkit [3], AprilTag [4], ArUco [5], ARtag [6] (for which the use in a rendezvous scenario has been proposed by Fiala [7]), etc. provide planar features. While these features are easy to produce, scaleable and have low complexity their reliability suffers from pose ambiguities and loss of precision under frontal observations. To mitigate these issues, often a protrusion or indentation is added to the marker system to make the features used in the pose determination algorithm non-planar. While this solves both issues stated above, it is not always feasible for ADR according to the stakeholder requirements found in this thesis.

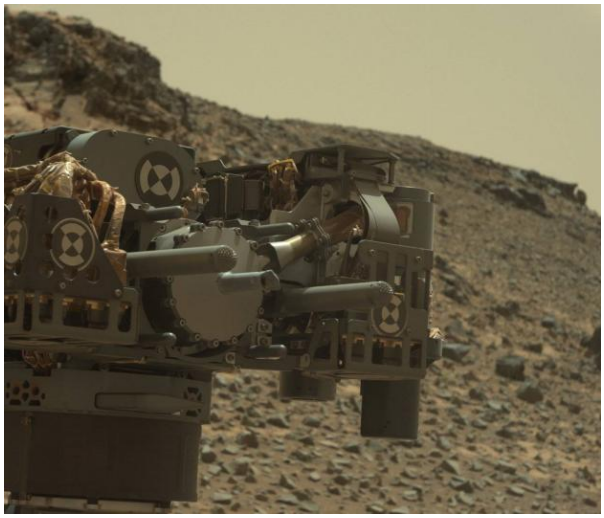


Figure 1.1: Fiducial markers on Curiosity. Image by NASA/JPL-Caltech/MSSS.

Alternatives to 3-dimensional fiducials have been posed in the form of encoding markers. These markers encode an extra dimension of information in the signal between marker and sensor, thus increasing the robustness of the fiducial system. However, existing solutions are hard to manufacture, qualify and scale. Therefore, a knowledge gap exists in the design of robust, planar fiducials for ADR.

1.2. Research Outline

The research objective of this thesis is:

To contribute to the development of a relative navigation system for Active Debris Removal by designing a marker system for relative spacecraft navigation within the context of Active Debris Removal.

To meet this objective, a system engineering approach is applied. In doing so, a number of knowledge gaps were identified. This thesis aims to fulfil the following knowledge gaps:

1. Robust, planar fiducial design for ADR.
2. Encoding markers for space applications.
3. System design of passive fiducials for space.
4. Virtual point encoding.

No robust and planar fiducials exist for ADR to the knowledge of the author. Little work has been done on encoding markers in general, and no work has been done on encoding markers for space applications. A

structured system design of *passive* fiducials (i.e. without the use of power) for space has to the knowledge of the author not been published. Finally, no existing fiducials encode a virtual point in space in their signal.

The system engineering approach employed in this thesis was applied to fulfil these knowledge gaps. Some definitions, such as the definition of planarity and robustness, are further developed in the requirement engineering for this work.

1.3. Research Questions

The scientific value of this work lies in the fulfilment of the knowledge gaps defined above. While this thesis consists of the design of a system, this system design was driven by the need to answer a set of defined research questions. Similar to the research objective, the main research question is defined as follows:

What is the most suitable marker system for relative navigation for Active Debris Removal?

Sub-questions that relate to this main research question have been defined as follows:

[RQ-1] What are appropriate high-level system requirements for a relative navigation system for ADR?

- How should a fiducial marker system be illuminated in an ADR scenario?
- What sensor for relative navigation should be used on the chaser spacecraft?
- How can a marker system be implemented at the lowest cost to the spacecraft owner?

[RQ-2] How can pose robustness be ensured while retaining planarity of the marker system?

- What failure modes are present in current planar marker systems?
- What causes these failures?
- How can these failures be prevented in a fiducial marker system for space?

[RQ-3] What is the sensitivity of the end-to-end system performance with respect to the main design parameters?

- What is the sensitivity of the end-to-end system performance with respect to the specifications of the chaser sensor?
- What is the sensitivity of the end-to-end system performance with respect to the marker size?
- What is the sensitivity of the end-to-end system performance with respect to the spacing of markers?
- What is the sensitivity of the end-to-end system performance with respect to the number of markers?

[RQ-4] What is the performance of the designed fiducial marker system?

- What is the relative performance of the fiducial marker with respect to existing solutions?
- What is the theoretical attainable performance of the fiducial marker system based on the sensitivity of the end-to-end system and the measured performance?

1.4. Thesis Structure

This thesis is structured as follows; First, in chapter 2, the necessary background to this thesis is presented. A literature survey is presented that details the problem of space debris and its political implications, rendezvous and the specific conditions concerning ADR, vision-based relative navigation, the Perspective-N-Point (PnP) problem and the challenges that arise when using planar targets in pose estimation. In chapter 3,

the system engineering approach utilised is extensively described and the requirements for RENDER are derived. The appropriate design options for the marker system concept are identified and an Analytic Hierarchy Process (AHP) is used to select the most appropriate concept in chapter 4. Following the concept selection, the fiducial design is presented in chapter 5. Since the research objective does not just cover a fiducial design but a higher level fiducial system design for ADR, the end-to-end system design is explored in chapter 6. Next, both the fiducial design is tested using a physical and a virtual experiment for which the experimental setups are detailed in chapter 7. The results for both experiments, as well as the result of the Monte Carlo simulation of the end-to-end system, are presented in chapter 8. Finally, concluding remarks and recommendations for future work are presented in chapter 9.

2

Background

To justify the design of a fiducial marker system for relative navigation for ADR, this chapter will describe the context of the problem which will provide the input for the system design.

For justification of some of the design choices made, the problem of space debris and its political implications needs to be addressed. This is covered in section 2.1. Since this thesis is concerned with a form of debris removal that requires rendezvous with another spacecraft, a brief overview of this subject is given in section 2.3. Next, the problem of vision-based relative navigation is defined and a system diagram for the end-to-end vision-based relative navigation system is presented in section 2.5. In section 2.6 the PnP problem is introduced and an overview of the state-of-the-art PnP solvers is given. Finally, some of the challenges that exist in using planar targets as fiducial for pose estimation are dealt with in section 2.7.

2.1. Space Debris

The United Nations Office for Outer Space Affairs (UNOOSA) defines space debris as: "all man-made objects including fragments and elements thereof, in Earth orbit or re-entering the atmosphere, that are non-functional" [8]. Objects can be as small as a flake of paint or as large as a defunct spacecraft the size of a school bus. While these objects are detrimental to the space environment, the incentives to remove them from orbit are small.

However, space debris is a growing threat to the access to space from Earth. This has tremendous implications for not just the space industry, but human progress in general. As the orbital space around Earth is becoming more littered with objects, the risk of impact with functional assets increases. Arguably, the dependence on space is increasing at the same time: science and exploration, Earth observation, communication, (environmental) policy making, defence, navigation etc. etc. are nowadays all dependant on a reliable access to space.

The problem that debris causes is clear: at the tremendous relative velocities of objects in space, even small and light objects have enormous destructive energy when impacting with a functional space asset. In a well-known paper by Kessler and Cour-Palais [9], the authors pose the possibility of the formation of a debris belt caused by on-orbit collisions. While the formation of an actual debris belt is unlikely due to atmospheric drag [10], the number of objects, especially in Low Earth Orbit (LEO), is increasing rapidly.

Space debris is caused by several factors: (improper) disposal, explosions, disintegration etc. However, the most concerning and prevalent event is the collision of existing objects [1, 9–11]. Where most causes of space debris can be mitigated by adhering to appropriate mitigation guidelines as proposed by the United Na-

tions [12] (and adopted by the European Cooperation for Space Standardization (ECSS) in ISO 24113:2011 [13] which applies to all ESA missions), collisions can only be prevented by detecting and manoeuvring or by lowering the number of objects in orbit.

In a paper from 2010, Kessler et al. provide further clarification on what has become known as the so-called "Kessler Syndrome" [10]. The Kessler syndrome describes the exponential growth of debris due to cascading collisions, or in other words: debris begets debris begets debris, *ad infinum*. Therefore, collisions are not only the cause but also the consequence of the problem of space debris. The rate of debris growth until 2010 was approximately linear at 300 objects per year [10]. However, when collisions become the predominant cause of debris in an orbital region this growth could quickly become uncontrolled and exponential, causing tremendous risks to space assets. Liou analysed the future increase of the number of objects in the "business-as-usual" scenario [14]. This study predicted that the rate of debris growth will be largest in Low Earth Orbit (LEO) and in the current scenario will become non-linear around 2050 as shown in Figure 2.1. In both Medium Earth Orbit (MEO) and Geostationary Earth Orbit (GEO) the number of objects and mass is smaller. Furthermore, the orbital volume is larger and debris mitigation measures are predicted to limit the rate of debris growth.

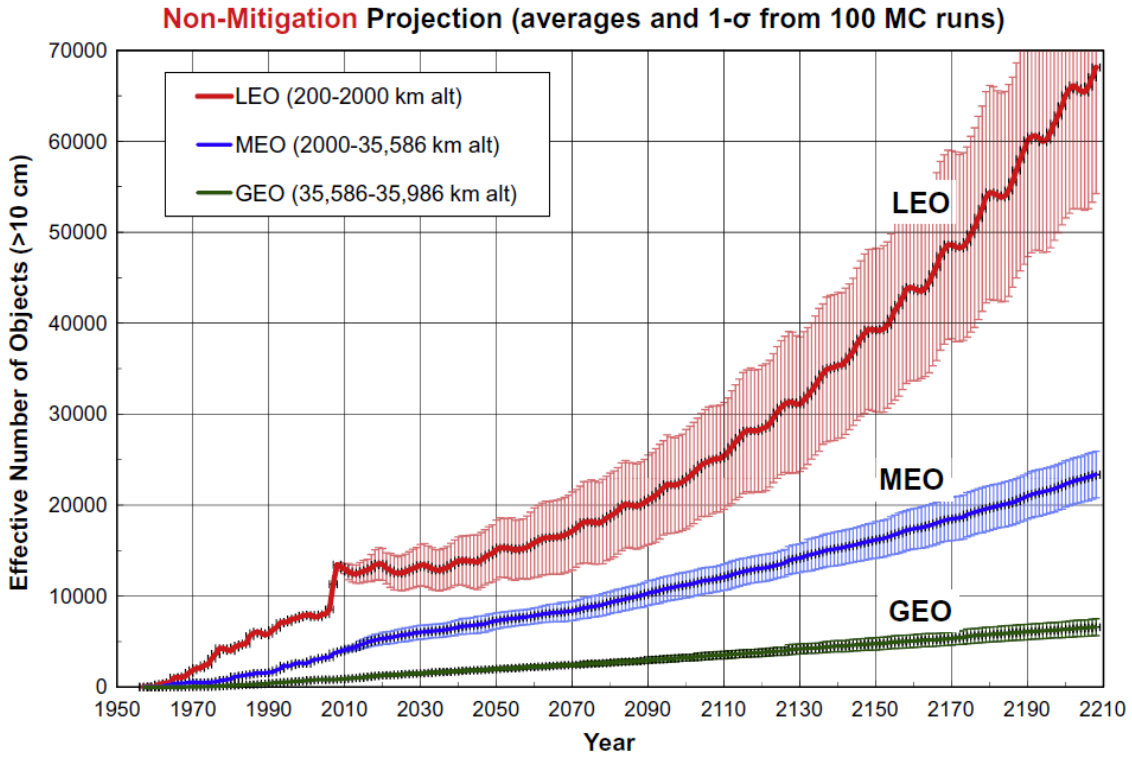


Figure 2.1: Predicted debris growth based on Monte Carlo simulation. Reprinted from [14], with permission from COSPAR

Risk can be calculated as a factor of likelihood times consequence. This is no different for space debris: the orbit, volume and mass of an object determine the consequence of collision for the space environment. A collision in a critical orbit is more consequential than a collision in a low-traffic orbit. Volume and mass are indicators of the amount of debris created and the impact force an object will have when a collision takes place. The likelihood of a collision taking place is dependant on the *debris flux*, i.e. the number of objects per volume. If the debris flux increases, so does the likelihood of a collision. Thus, this increases the risk to spacecraft in orbit.

Consequently, with the rising risks to current assets and future exponential growth of debris, it is clear that action should be taken to limit the growth of space debris. It is also clear that even if debris mitigation measures were to be successfully implemented in 100% of all new space missions, this would not be enough to limit the growth of objects in LEO due to the present object density [10]. Furthermore, full adherence to mit-

igation guidelines is neither politically nor technically feasible. Thus, it is essential to perform Active Debris Removal (ADR). According to existing studies, between 5 and 10 objects should be removed every year to keep the debris environment similar to 2010 levels [14, 15].

The effectiveness of ADR strongly depends on the orbit, volume and mass of the removed object. An emphasis should be placed on removing the most critical objects in orbit, as this would greatly increase the impact ADR can have on the orbital environment [10]. A good candidate is the removal of rocket bodies since these have simple shapes and carry no sensitive technologies [14]. In any case, ADR should be first and foremost focused on LEO since this is the most critical environment.

Multiple ADR techniques exist. An overview is provided by Shan, Guo and Gill in [2] and by Mark and Kamath in [16]. While some techniques are ground-based (such as the use of a laser system to slow an object down), most ADR techniques are space-based and involve a rendezvous scenario with the target object. It is these scenarios for which this thesis fills a critical knowledge gap.

2.2. Political Implications of Space Debris and Active Debris Removal

The problem of space debris is similar in nature to the Ozone layer thinning (which was most prevalent in the last century) and the current problem of global warming. Both these issues have a known and arguably preventable cause which is human-made. However, liability and accountability is a large hurdle to overcome if these problems are to be countered. The international, effective cooperation on the Ozone layer problem shows that these problems can be solved. With international consensus and effective political action, the gasses that caused this issue were essentially banned and an effective control mechanism was implemented. A similar solution is necessary for the space debris issue.

However, this is not an easy task. The current measures for the protection of the space environment are non-binding and voluntary. Any economic incentives to perform both mitigation and ADR are indirect: there is an added benefit of not worsening the environment but no direct payment for a successful debris mitigation strategy. Pelton has proposed a solution in the form of a fund for debris removal in [17]. A mandatory contribution from stakeholders in space missions under national or international law to such a fund could finance (commercial) ADR missions and technology.

Many of the technologies involved in performing ADR could be regarded as military assets or weapon systems [18]. If a functioning spacecraft from one nation is removed by another nation, this could be considered an act of aggression. Furthermore, the problem is shared among all space-going nations (and arguably all nations), indicating that it is in the interest of all to develop and deploy the necessary technologies. It is clear that international cooperation and transparency in both the development as well as deployment of these technologies is essential.

ADR can be facilitated from a system engineering perspective not only by building the best suitable system for removal but also by creating systems that are most easily removed. The international community should commit to not only mitigation measures and removal technologies but also to facilitating the removal of future spacecraft (here spacecraft should be considered broadly, also including for example rocket stages). This is called Design for Removal (D4R). D4R would ensure the simplification of removal and increase the reliability of ADR missions. As with the current CubeSat developments, standardisation would greatly benefit these missions, bringing down costs and complexity.

It can therefore be concluded that any system that would aim to facilitate D4R should be as universally applicable as possible while keeping the incentive for using the system higher than its costs. If the space debris problem is to be tackled effectively, given the fact that nations cannot impose debris mitigation regulations forcefully on other nations, the technical solutions should be natural to implement for all due to as low as possible cost of implementation (in terms of financial resources, but also system resources such as volume, power, mass etc.).

2.3. Rendezvous

A rendezvous can be defined as the approach of an object in orbit by a spacecraft, usually for the purpose of docking with that object. Orbital dynamics dictate the relative motion between the two objects, making these manoeuvres arguably complicated. There are many scenarios where a rendezvous between two spacecraft is necessary, for example: capturing, OOS, monitoring or inspecting [19].

A typical rendezvous mission can be divided into 4 main phases [20]:

1. Launch of chaser spacecraft and insertion into target orbital plan using absolute navigation. Detection of target.
2. Approach of target orbit using orbital transfers, absolute navigation.
3. A series of impulses brings the chaser closer to the target at a defined distance using relative navigation. The nature of these manoeuvres (V-bar, S-bar) depends on the mission.
4. Finally, a forced translation is performed to dock the chaser with the target.

Woffinden and Geller provide a historical overview of the rendezvous capabilities of the US and Russia up to 2007 in [19]. The first rendezvous was performed manually by Neil Armstrong in 1966. The year after, in October 1967, the Russian Kosmos 186 and Kosmos 188 spacecraft performed the first automated rendezvous. Relative navigation for most of these early spacecraft was relying on radar systems (and in the case of piloted spacecraft, on human eyesight). These radar systems were active systems on both the chaser and target side. With the advent of the Spaceshuttle in 1983, some advances in rendezvous capabilities were made. The Spaceshuttle was able to perform both manual and (partially) automated rendezvous manoeuvres and had apart from a radar system also an Inertial Measurement Unit (IMU), some optical systems and laser ranging equipment. On the Russian side, the so-called Kurs system is currently used in the Soyuz vehicles that visit the ISS. These radar systems have a strong heritage, but are power-intensive and have a relatively high mass. Furthermore, these systems mostly require an active target.

More recently, navigation systems used on spacecraft such as the ESA Automated Transfer Vehicle (ATV), dedicated experimental flights and Spaceshuttle flight experiments (described extensively in the literature survey of this thesis [21] and in [22–26]), rely on optoelectrical systems usually in conjunction with active or passive marker systems on the target. Some of these systems can perform pose estimation on a "non-prepared" spacecraft, i.e. having no dedicated fiducials. This requires feature extraction of the natural shape and features of the spacecraft [27]. For absolute as well as relative navigation, in certain orbits Global Positioning System (GPS) may be used [19]. Lastly, Radio Frequency (RF) systems have been used for relative navigation in the PRISMA mission [28].

For rendezvous manoeuvres, relative navigation systems based on optoelectrical sensors (infrared, visual etc.) have several benefits. These systems can be used with unprepared target objects as well as passively and actively prepared target objects (having unpowered or powered fiducials such as LED's). Furthermore, they are relatively low cost, have lower complexity and volume and mass than other systems such as RF systems and radar. Finally, the achievable accuracy is suitable for rendezvous and has been validated through historical missions [21].

2.4. Rendezvous Conditions for ADR

To define system requirements for a relative navigation system for ADR, it is necessary to define the environment in which such a system is expected to function. The rendezvous conditions for ADR provide a more complex scenario than any previous missions.

In [29], Šilha et al. studied the rotational rate of space debris based on photometric measurements on 189 objects in LEO. In LEO, most of the observed objects in this study (around 70%) are stable. 26.5% of the

present objects are slow rotators and around 3% are rotators. The authors define slow rotators as objects that have a rotational rate that is longer than the duration of observation (for LEO between 3-12 minutes) whereas rotators have a faster rotational rate than the observation period. The relatively low number of rotators in LEO can be explained by the damping effects of the atmospheric drag and eddy currents due to the Earth magnetic field. For the non-functional satellite ENVISAT, an important candidate object for debris removal, the observed rotational rate in 2018 was 1.96°/s. This rotational rate has decelerated from a rate of around 2.67°/s in 2013 [30].

Deloo and Mooij studied illumination conditions of ADR for LEO in [31]. These conditions are challenging: a short orbital period of between 90 and 100 minutes can cause highly variable lighting conditions which cause issues for power generation (in eclipse) and relative navigation systems that require a level of illumination. Furthermore, the direction of sunlight changes rapidly. A worst-case scenario for any electro-optical relative navigation system is blinding by sunlight. Consequently, due to the uncertain nature of space debris dynamics and the diverse illumination conditions, a relative vision-based navigation system should deal with a wide range of illumination scenarios.

2.5. Vision-Based Relative Navigation

Relative navigation is the continuous determination and filtering of the 6 Degree Of Freedom (DOF) pose of the target spacecraft [32]. The is shown in Figure 2.2. Vision-based navigation involves a chaser system that uses an electro-optical sensor. While the relative (rotational) velocities between chaser and target are required for accurate relative navigation, these are the result of an integration of the 6-DOF state measurements.

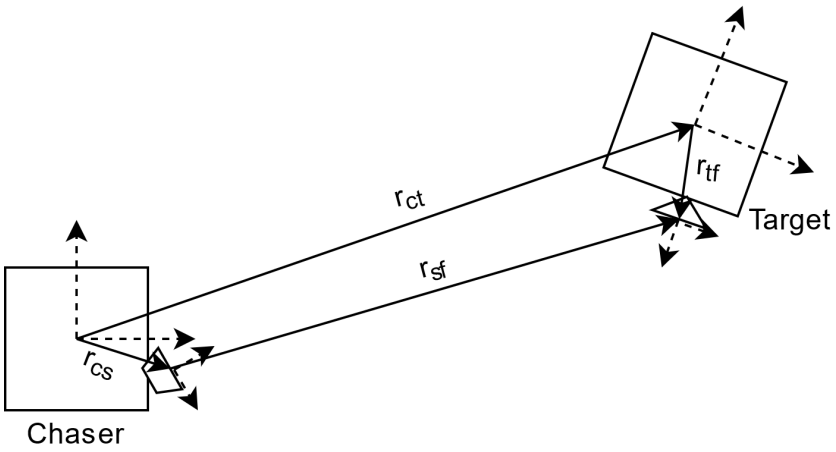


Figure 2.2: Schematic representation of the relative navigation problem

The relative pose consists of the translation (depicted in Equation 2.1) and rotation (depicted in Equation 2.2) between chaser and target. These can be combined in a homogeneous transformation matrix shown in Equation 2.3. If a prepared target is used, which includes a fiducial, the relative navigation system typically determines the relative pose of the fiducial relative to the sensor on the chaser. Then r_{ct} and R_{ct} can be determined by using the known transformations between the chaser reference frame and sensor (T_{cs}) and target reference frame and fiducial (T_{tf}). This is shown in Equation 2.4.

$$r_{ct} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.1}$$

$$R_{ct} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.2}$$

$$T_{ct} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$T_{ct} = T_{cs} * T_{sf} * T_{tf}^{-1} \tag{2.4}$$

The target in relative navigation can be *prepared* or *unprepared*. The former has a fiducial system that enables robust feature extraction and pose determination, the latter does not. Furthermore, these fiducials can be divided into *actively* or *passively* prepared. The former system requires power to function, the latter does not. A distinction between chaser systems that utilise 3D measurements (i.e. LIDAR or stereo-cameras) and monocular cameras can be made [32]. If costs, mass, volume, power consumption and complexity are to be kept at a minimum, the monocular sensor system should be preferred at the cost of some robustness of the pose estimation [27].

Several monocular pose estimation methods exist: model-based (appearance-based or feature-based) and Convolutional Neural Network (CNN)-based [27]. The appearance-based methods are based on the comparison of the captured spacecraft appearance to an offline image database. CNN-based methods utilise novel computational methods that provide lower computational complexity, but may lack robustness due to the difficulty in acquiring representative training data and the challenging illumination conditions of the space environment. The most robust method is therefore arguably the feature-based pose estimation, further explained in the next section.

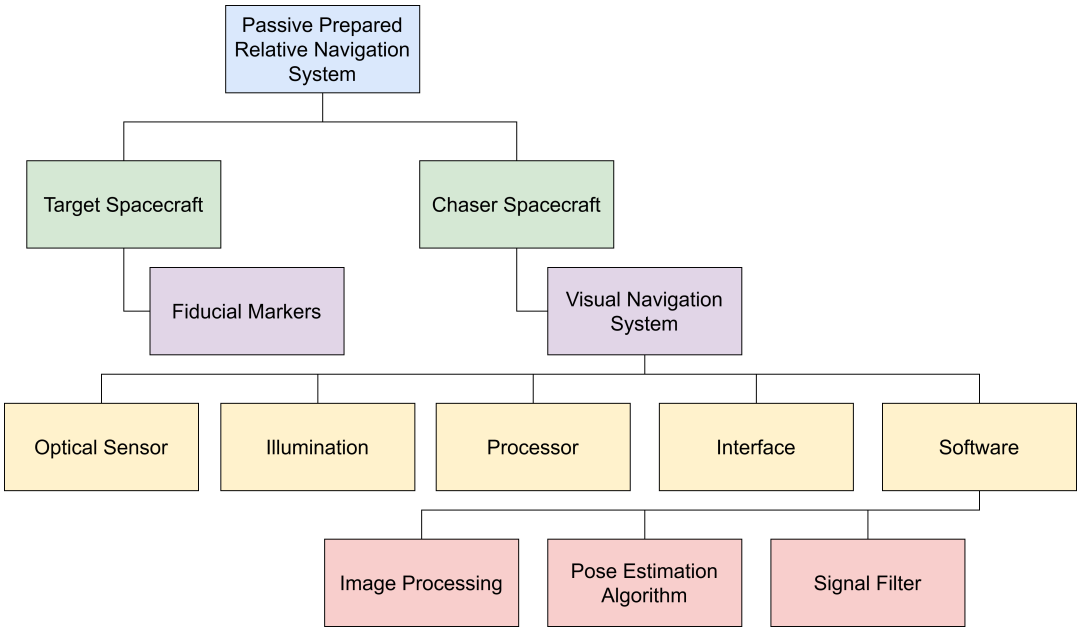


Figure 2.3: System diagram for vision-based relative navigation

A system overview of the end-to-end vision-based relative navigation system is shown in Figure 2.3. It shows the interdependencies of the different subsystems. While an illumination block is shown in this system concept, it should be noted that this is optional depending on the selected marker taxonomy. This thesis is focused on the left (target) side of this diagram. However, the system cannot be designed nor verified without defining the chaser side of the end-to-end system.

2.6. Feature Based Pose Estimation: the PnP Problem

As stated, the problem of pose estimation using a monocular camera entails finding the correct transformation matrix between the sensor frame and the target frame. When this pose estimation is performed based on features present on the target, this is called *feature based pose estimation.* If these features are purposefully applied to the target, these features are called *fiducials.* A fiducial may include multiple features (e.g. Aruco markers) or a single feature (e.g. retro-reflectors). The benefits of using a fiducial on the target are more robust feature extraction and the known location of features in the target frame. Furthermore, under certain circumstances, a fiducial can provide an additional dimension of information (as explained in chapter 4).

While features from a fiducial may be used directly in the navigation filter of the end-to-end relative navigation system, an initial pose estimate is usually required [27]. This estimate for the 6-DOF relative pose requires a certain minimum number of features. Fischler and Bolles have provided a well-known formal definition of the pose estimation problem using features or control points:

"Given the relative spatial locations of n control points, and given the angle to every pair of control points from an additional point called the Center of Perspective (CP) find the lengths of the line segments ("legs") joining the CP to each of the control points. We call this the "perspective-n-point" problem (PnP)." [33]

It should be noted that a central assumption for the definition of this problem is that the camera is calibrated, i.e. the the intrinsic camera parameters are known (camera calibration is covered in chapter 7. The output of the PnP algorithm is the rotation and translation of the object with respect to the camera. This transformation can be used to back project the object points on the image plane using the following equation:

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.5}$$

Here, s is a scale factor, u and v are the coordinates of the respective points in the image frame, f_x and f_y represent the focal length in terms of pixels, c_x and c_y represent the coordinates of the principal point. This is represented in Figure 2.4.

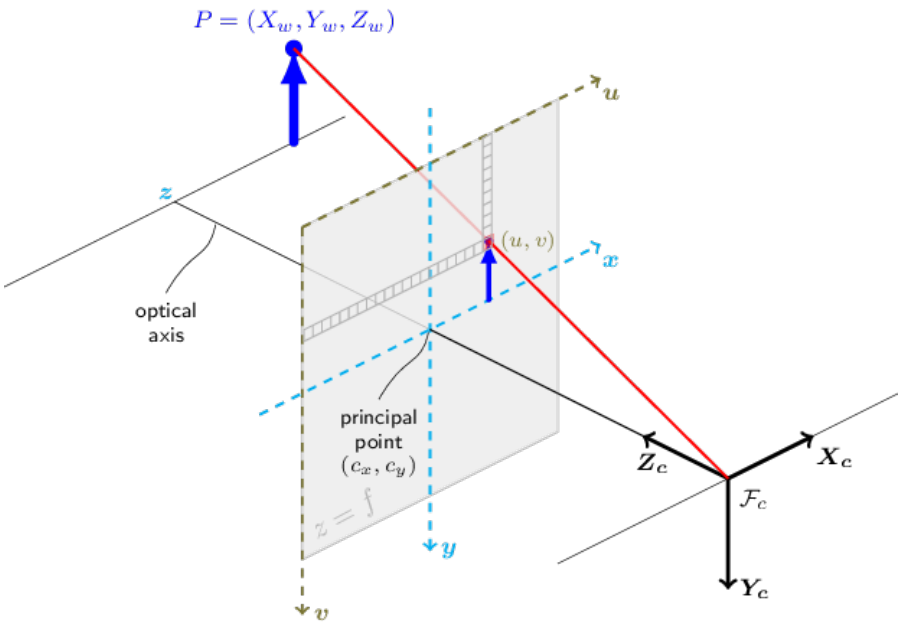


Figure 2.4: Pinhole Camera Model as applied to the PnP problem. Reprinted from [34], under the Apache License.

The pose, as defined by the transformation matrix in Equation 2.3, consists of 6 unknowns: 3 unknowns for the relative attitude, and 3 unknowns for the relative position. Therefore, with 3 recorded points on the image plane corresponding to 3 points in the target frame this equation becomes solvable in an ideal situation. However, in a real-world application, several problems persist. Firstly, as Fischler and Bolles show in [33], the so-called P3P problem has at most 8 solutions out of which 4 lie in front of the camera (and are thus physically feasible). This can be explained by Equation 2.6 corresponding to Figure 2.5, where it is clear that the 3 independent equations with 3 unknowns of the second-order can have at most 8 solutions.

$$\begin{aligned}(R_{ab})^2 &= a^2 + b^2 - 2 * a * b * cos(\theta_{ab}) \\ (R_{ac})^2 &= a^2 + c^2 - 2 * a * c * cos(\theta_{ac}) \\ (R_{bc})^2 &= b^2 + c^2 - 2 * b * c * cos(\theta_{bc})\end{aligned} \tag{2.6}$$

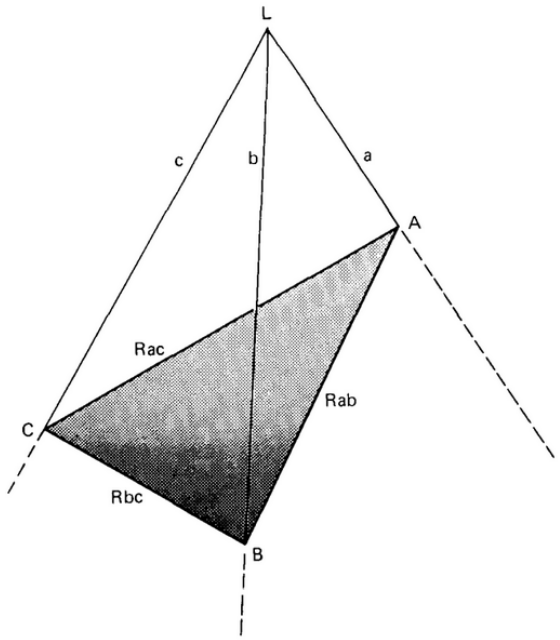


Figure 2.5: The P3P problem. L indicates the center of perspective in this image. Reprinted from [33], with permission of the Association for Computing Machinery.

This problem can be solved directly by transforming these equations to a polynomial, for which Haralick et al. survey a number of solutions [35]. While iterative solutions for the P3P problem exist, these require an initial guess of the pose which is usually not available in computer vision. The direct solution of the polynomial as posed by Fischler and Bolles can provide 4 real solutions. A fourth co-planar point is needed at least in order to remove this ambiguity. Alternatively, 6 generally spaced points will always provide a unique solution [32, 33].

The algorithms that are used to solve this problem are called PnP solvers. While numerous solvers exist, some of the most common in the context of relative navigation for space are: PosIt, SoftPOSIT, the DLS-method and the EPnP [27, 32, 36].

PosIt [37] uses the assumption that the image points are obtained using a scaled orthographic projection. The algorithm thus finds an initial pose using Pose from Orthography and Scaling (POS). Next, the algorithm calculates the scaled orthographic projection of the feature points of the object in the found initial pose and recalculates the pose of these shifted points. This is iterated several times until an accurate pose is found. SoftPOSIT [38] utilises the same algorithm in the calculation of the relative pose but is also capable to determine point correspondences between image frame and object frame using the softassign algorithm [39]. The Direct Least-Squares (DLS) method [40] utilises a nonlinear least-squares cost function to compute all solutions of the pose estimation problem. These solutions are minima of the cost function which are found

analytically. The algorithm does not require a pose initialisation. The authors compare their algorithm to a number of state of the art alternatives and an iterative pose estimation scheme using a Levenberg-Marquardt minimisation of the sum of the squared reprojection error initialised by the DLS algorithm. The iterative pose estimation produces more accurate results at a higher computational cost. In its not iterative implementation, the size of the cost function scales with $O(1)$, making it feasible for use with a large number of points. The EPnP algorithm [41] is a non-iterative solution that scales with $O(n)$. The algorithm achieves this by expressing the points in the world frame as a weighted sum of four control points. Then, a constant number of quadratic equations is solved for the relative pose of these points. The authors propose using the EPnP algorithm in combination with a Gauss-Newton iteration scheme to maximise the accuracy of the pose estimation, which increases the runtime only marginally with respect to the non-iterative implementation.

2.7. Challenges of Pose Estimation Using Planar Targets

While most of the pose estimation algorithms work with planar targets, i.e. targets with co-planar feature points, these targets inherently produce perspective projections that are mathematically very similar. This can cause the pose estimation to fail. In addition, under frontal observations the perspective projection of planar targets can vary minimally, causing the pose estimation algorithm to lose accuracy. Both these issues are more severe under the influence of noise on the feature point locations. This section investigates the causes and effects of these issues on the pose solution using planar targets.

Geometrically, pose ambiguity is shown in Figure 2.6. When only the depicted co-planar features are provided as input to the PnP algorithm, both the pose depicted in green as well as the pose depicted in red is a valid solution. However, only one of the two poses can be correct in a physical sense.

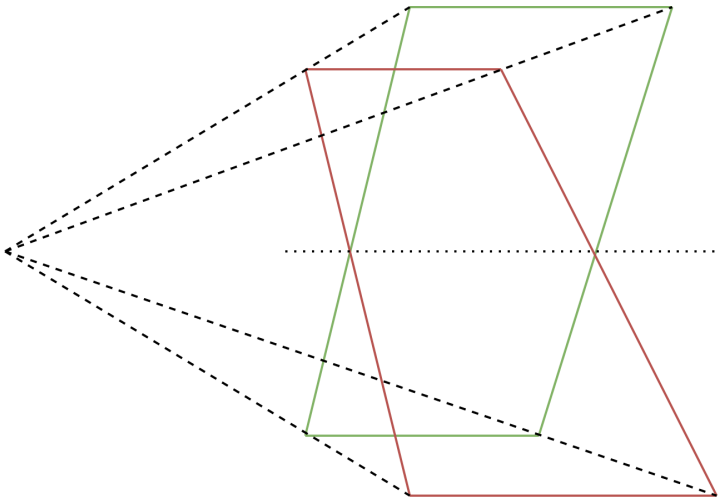


Figure 2.6: Pose ambiguity in planar feature points.

Oberkamp, DeMenthon and Davis deal with this problem in their adaption of POSIT for co-planar targets [42]. Their algorithm is designed such that it refines up to two possible solutions iteratively and provides a quality measure of the solutions which is based on the reprojection error. However, according to Schweighofer and Pinz, the perspective projection can only be approximated by the scaled orthographic projection used in POSIT when the distance of the camera is large with respect to the depth in the scene [43]. Schweighofer and Pinz show that pose ambiguities exist in many cases, including cameras with large FOV at close range. If the pose solution is found by minimising some error function, this ambiguity expresses itself in the existence of up to two local minima in the error function (shown in Figure 2.7). The authors show that with increasing distance between camera and target, the difference in absolute value between the error of the correct pose and its ambiguous counterpart decreases since the effect of perspective projection decreases. Similar to [42], the authors propose an iterative algorithm that finds both minima and uses the error function to estimate which is the global minimum. However, this approach can still cause problems if the absolute values of the minima are closely together due to weak perspective projection effects.

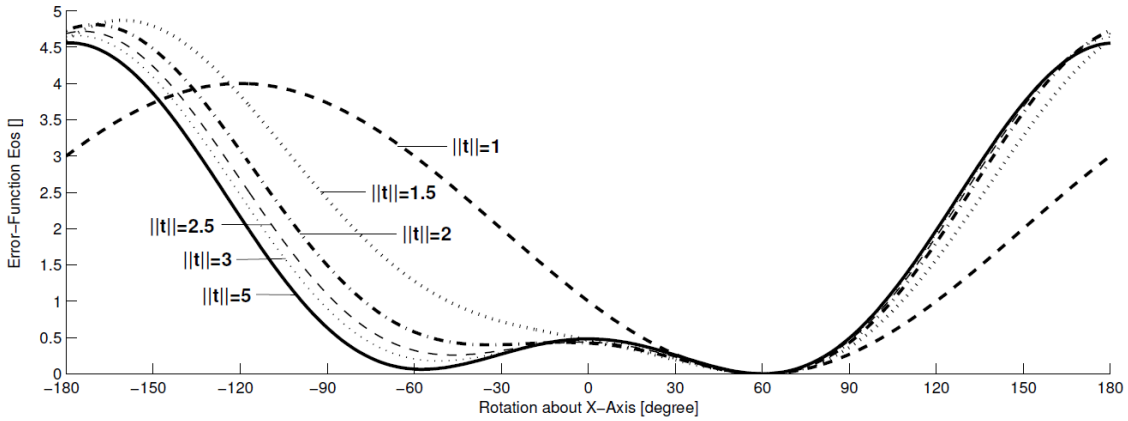


Figure 2.7: Error function plot for different camera to target distances $||t||$ (dimensionless) for a pose of 60 degrees. This plot assumes no noise on the input of the error function and uses the *object-space* error function [44] to calculate the error. Reprinted from [43], with permission. ©2006 IEEE.

In addition to pose ambiguity, the pose solution under frontal observations of planar markers is subject to performance loss due to the lack of perspective projective effects [45]. This can also be seen in an accuracy study of the planar ARToolKit fiducial in [46]: the fiducial has the lowest accuracy and precision at a camera angle of between 0 and 30 degrees.

A non-planar marker, such as the Space Vision Marker System (SVMS) depicted in Figure 2.8, does not suffer from either of these issues. The protrusion of the marker adds a feature point that solves the ambiguity due to an out-of-plane position in the world frame. Furthermore, the projection of this feature point moves by a larger distance on the image plane per degree rotation of the marker than the planar feature points. However, non-planar fiducials use significantly more system volume which may not be feasible in some applications.

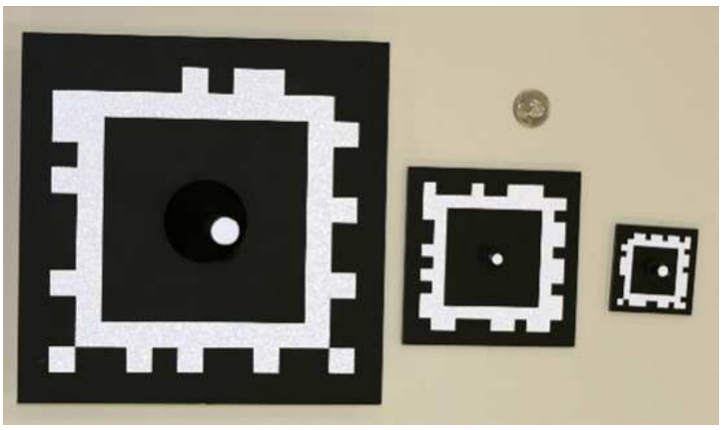


Figure 2.8: Space Vision Marker System (SVMS). This non-planar marker designed for relative navigation in space does not suffer from pose ambiguity nor from a loss of performance under frontal observations due to the increased perspective projection effects of the protrusion. Reprinted from [47], with permission from the author.

2.8. Conclusion

Space debris is a growing threat to human progress and requires ADR in order to stem the growth of objects, especially in LEO. To be most effective, ADR should focus on removing large and heavy objects from the most critical orbital trajectories. Given that it is currently impossible to impose forcefully international regulations on other nations, any effective system that aims to facilitate D4R should come at the lowest cost possible in terms of system resources such that all spacecraft owners are incentivised to implement the system.

Space rendezvous is the approach of an object in orbit by a spacecraft. For ADR the rendezvous conditions can be challenging due to rotating targets and unfavourable illumination conditions. This approach requires relative navigation which historically has been performed by humans, radar systems, GPS systems or RF systems. More recently, vision-based systems have been used to facilitate relative navigation. These systems have the advantage of being applicable to non-active targets (i.e. not requiring any power). Of these systems, the monocular sensor system requires the lowest amount of system resources.

In the passively prepared vision-based relative navigation system, applicable to the ADR problem, a feature-based pose estimation method is used. This method utilises a PnP solver to solve for the relative pose. However, if planar targets are used, these solvers suffer from pose ambiguity and loss of performance in frontal observations. A non-planar marker with a protrusion solves these issues, but at a cost of higher system volume.

3

System Engineering Approach and Requirement Engineering

The development of a fiducial marker system for ADR requires a systematic system engineering approach. To make sure that the system design does indeed fulfil the envisioned need and does so in the most optimal form (i.e. at the lowest cost and risk [48]), a thorough development of requirements is made. These requirements are used as input to the design process and for the identification of feasible design options in chapter 4.

The approach used to derive the system requirements is based on a structured system engineering process. This approach has been based largely on the approaches described in [48–51]. However, this approach has been adapted to suit the RENDER project. Since the system boundary for RENDER is more narrow than an end-to-end space mission, the number of system components is smaller and the system design can be started at a relatively lower level.

In this chapter, section 3.1 explains the system engineering approach used. According to ECSS-E-ST-10-06C [52], the process to establish the technical requirement specifications in a Phase 0 project starts with the identification and capture of user need or mission statements as well as environments and constraints. To this end, in section 3.2, the need and mission statement are derived in order to express the customer expectations. In section 3.3 the stakeholder analysis that identifies all the relevant stakeholders for RENDER is detailed. In the subsequent section, section 3.4, a complete set of stakeholder requirements is derived. These requirements are used to identify the system context and system concept of operations (section 3.5). Then, the system requirements are presented including their rationale, traceability, priority and verification method in section 3.6. Finally, the Monte Carlo simulation used to derive the accuracy and precision requirements is detailed and validated in section 3.7.

While the work here is presented linearly, it was conceived using a highly iterative approach. At each iteration, the design at that stage was analysed, tested, verified and adapted. The iterative nature and approach to the right side of the Vee model, as shown in Figure 3.1, i.e. the design synthesis, are further explained in chapter 4.

3.1. System Engineering Approach

The system engineering approach adopted in this thesis is based on the Vee life cycle model, as explained in the International Council on Systems Engineering (INCOSE) system engineering handbook [51]. This model shows the iterative nature of the system engineering process and the duality between decomposition of the problem with the design synthesis in the design phase. In Figure 3.1 the Vee model is adapted for this thesis.

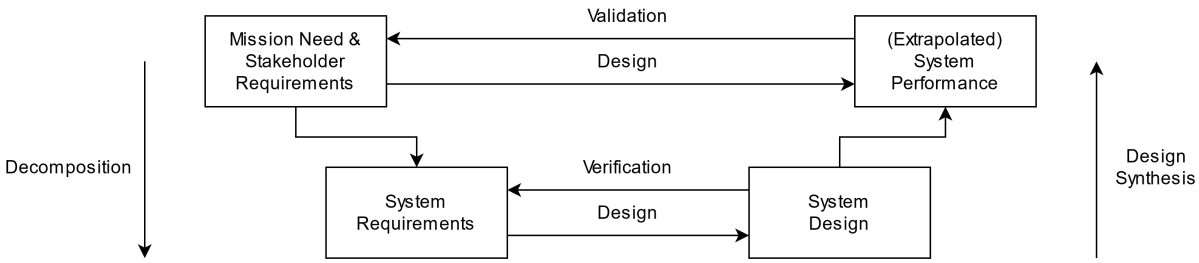


Figure 3.1: Vee Model Adaption for RENDER

In both levels of abstraction depicted in Figure 3.1 a feedback link between the left and right side of the Vee model is established. On the left side of the Vee model (i.e. system decomposition), stakeholder requirements feed into system requirements, but also provide the input to the system performance and benchmark for the validation process. One level lower, system requirements in the design synthesis side of the Vee model feed into the system design. For the verification of the system design, the system requirements are used.

Note that contrary to the Vee model used in end-to-end space missions, a third level (i.e. subsystem level) is not defined. The reason for this is two-fold: firstly, as stated in the introduction of this chapter, the system boundary for RENDER is more narrow than an end-to-end space mission. Secondly, the system design can be considered a Phase 0 study. Deriving detailed subsystem requirements is cannot be justified at this stage in the design process, due to the large number of unknown unknowns. It is therefore deemed more applicable to define the system up to a certain level of detail and leave flexibility in terms of e.g. system dimensions, materials, etc.

As stated above, the right side of the Vee model (i.e. the design synthesis) is an iterative process. This involves synthesising requirements into a design, verifying this design according to the requirements and iterating accordingly. The design synthesis process is structured in a bottom-up fashion. First, a design is made for the fiducial marker system, and this system is verified against the system requirements. Then, at the highest level of the design in the Vee model of Figure 3.1, an analysis is made on the application of this marker in an end-to-end system. These results allow validation against the stakeholder requirements and the mission need.

3.2. Need and Mission Statement

In order to express the goal of the system and the need it is aimed to fulfil, a need and mission statement are derived. These statements capture the why and how behind the system development, and provide a starting point for deriving the customer requirements The need statement is as follows:

Space debris is an active threat to the future of spaceflight and can only be reduced by active debris removal. Future spacecraft need to be prepared for relative navigation to facilitate active debris removal.

This statement encapsulates the need for a system that facilitates relative navigation by a potential future target spacecraft. This is further developed into the mission statement of RENDER:

RENDER will provide a fiducial marker system for active debris removal, applicable to a broad range of future spacecraft.

In this mission statement, *fiducial marker system* is defined as a system that provides a distinctive signal to a chaser spacecraft that allows robust pose estimation of the object by this chaser spacecraft. It should be noted that while the mission statement uses the term *spacecraft*, this term should be considered broadly here. The fiducial marker system could also be applicable to rocket stages, launch adaptors, etc. Furthermore, the system should be *broadly* applicable. As established in section 2.2, an effective fiducial marker for ADR would benefit greatly from standardisation.

3.3. Stakeholder Analysis

With the need and mission statement established, a stakeholder analysis is performed to be able to derive stakeholder requirements. Before these stakeholder requirements can be derived, it is necessary to identify all relevant stakeholders and classify these stakeholders as *(key) active* or *passive*.

Firstly, ESA is considered the customer for RENDER. ESA is therefore identified as key active stakeholder as it is the stakeholder that commissioned the system.

However, as explained in the previous section and captured in the mission statement, the system is intended to be broadly applicable. It should therefore be noted here that not only be applicable to spacecraft launched by ESA, but also to most other future spacecraft. Therefore, even though no such stakeholder is (at the current stage) actively involved in the design of this system, a future target spacecraft owner is identified as an active stakeholder. This stakeholder could be a commercial entity or even another space agency that is bound by future debris mitigation regulations to include a fiducial marker system on a spacecraft. These spacecraft owners should be incentified to use the system, or at the very least not be discouraged to implement it (by for example high mass or volume).

A third stakeholder is identified as the future chaser spacecraft owner. This chaser spacecraft will house the active part of the relative navigation system, for which the target system is a key driver of the system requirements. This stakeholder is an active stakeholder. This stakeholder would benefit from a target system that improves relative navigation performance at the lowest cost to the chaser spacecraft.

A fourth, passive, stakeholder is the combined international space debris mitigation guidelines. These guidelines dictate the mitigation techniques applied to space missions, and therefore passively influence the success of the marker system. These guidelines would be more enforceable if the technical solutions that underpin the guidelines are low cost, flexible, and allow for standardisation.

3.4. Stakeholder Requirements

Having identified the stakeholders, the stakeholder requirements are derived. The following stakeholder requirements are identified for the fiducial marker system:

[STA-REQ-1] (Capability) The system shall enable relative navigation for typical ADR missions utilising robotic capture between a chaser and target spacecraft. *Rationale:* This stakeholder requirement captures the research objective of this thesis and the need identified by the customer.

[STA-REQ-2] (Characteristic) The system shall be minimally invasive to the target spacecraft it is applied to. *Rationale:* All identified stakeholders with the exception of the future chaser spacecraft owner need a system that requires a minimal amount of resources (i.e. mass, volume, footprint). The application of a relative navigation aid system is not currently not enforced by guidelines, thus the incentive to use such a system should be intrinsic and at essentially nil costs to the spacecraft owner (the customer or a future target spacecraft owner. Furthermore, if these markers are ever to be made mandatory on future spacecraft by international guidelines, the implementation should be as minimally invasive as possible in order to be realistic. Therefore, this stakeholder requirement captures the need of the customer, future target spacecraft owner and international space debris mitigation guidelines.

[STA-REQ-3] (Characteristic) The system shall be unpowered. *Rationale:* Since the system is to be applied to spacecraft in order to enable their removal when it has failed to do so autonomously, and since the inclusion of a power source increases the risk to the spacecraft the system needs to be unpowered. This stakeholder requirement captures a customer need.

[STA-REQ-4] (Capability) The system shall function for at least 20 years. *Rationale:* This stakeholder requirement ensures that the system is able to function in the space environment for an adequate du-

ration. This enables a wide enough time frame for removal of a target spacecraft. This stakeholder requirement captures a need of the customer and a future chaser spacecraft owner.

[STA-REQ-5] (Characteristic) The system shall use only space-grade materials. *Rationale:* In order to minimise the risk to the target spacecraft, the materials shall use non-dangerous materials that are robust to the environment. This stakeholder requirement captures a need of the customer, a future target spacecraft owner and international space debris mitigation guidelines for similar reasons as STA-REQ-2.

[STA-REQ-6] (Capability) The system shall be universally applicable to target spacecraft regardless of size class, excluding 12U CubeSat and smaller spacecraft. *Rationale:* The research objective is not limited to a specific spacecraft class. Therefore, the design of a visual navigational aid needs to be inherently flexible in terms of application. Since the constraints of an ADR mission are currently undefined, a marker that fulfils the need of this hypothetical mission is applicable to most target spacecraft. CubeSat and smaller are not included, since these systems are not likely candidates for ADR. This stakeholder requirement captures a need from international space debris mitigation guidelines.

[STA-REQ-7] (Characteristic) The end-to-end relative navigation system shall be low-cost. *Rationale:* The applicability of the system is increased by keeping it low-cost. This is true for both the marker system and the end-to-end system, including the chaser spacecraft. Since the large majority of the end-to-end system costs and complexity are at the chaser side of the end-to-end system, this requirement will limit the chaser sensor options. This stakeholder requirement captures a need from the customer, future target and chaser spacecraft owners.

[STA-REQ-8] (Capability) The marker system shall impose requirements on the active chaser component of the end-to-end navigation system that can be fulfilled by current off-the-shelve components. *Rationale:* A future chaser spacecraft shall use the marker system as reference fiducial by using a feasible sensor in terms of e.g. resolution, illumination, power, wavelength and FOV. This stakeholder requirement expresses a need from the customer and future chaser spacecraft owner.

[STA-REQ-9] (Characteristic) The marker system shall be planar. *Rationale:* In addition to STA-REQ-2, the fiducial marker system needs to be as flat as possible to prevent occlusions, blocking of (robotic) movement and/or FOV of instruments. This requirement expresses a need from the customer.

3.5. Concept of Operations and System Context

A concept of operations is an expression of the stakeholder expectations and is a tool that can be used to validate scope and system boundary. In the context of RENDER, it is useful to define a concept of operations for the ADR mission type the system shall be designed for.

While several contactless methods for ADR exist, the majority of ADR concepts require a form of contact as well as robotics [53]. The concept of operations for RENDER is based on a mission that would require both contact and robotics, as is required by the customer in [STA-REQ-1]. Colmenarejo et al. define a typical approach for an ADR mission in [53]. The authors define 4 mission phases: P0 - far distance approach, P1 - final approach, P2 - capture, P3 - disposal. A key assumption, in consultation with the customer, is that the RENDER system applies to (part of) the P1 and P2 phase of an ADR mission. Before the RENDER system is utilised for the relative navigation, the Guidance, Navigation and Control (GNC) system performs the relative navigation task.

In Figure 3.2 an applicable rendezvous scenario for ADR is shown, based on [53]. The letters indicate different steps of the approach. A indicates a drift orbit 400 metres below V-bar, followed at B by a drift orbit 50 metres below V-bar. At point C, a safe orbit is held for a several orbits. After this, at 100 metres or point D in Figure 3.2, a hold-point on the V-bar is held by the chaser. Then a forced motion approach over V-bar up to 30 metres is made at which proximity operations begin. These operations depend on the ADR mission, but it is assumed in this thesis that relative navigation is performed without the fiducial marker system as it is proposed here. Then, another forced motion is performed to 20 metres. At 20 metres, station keeping and debris characterisation are commenced. For this use case, RENDER should facilitate the characterisation of the debris in

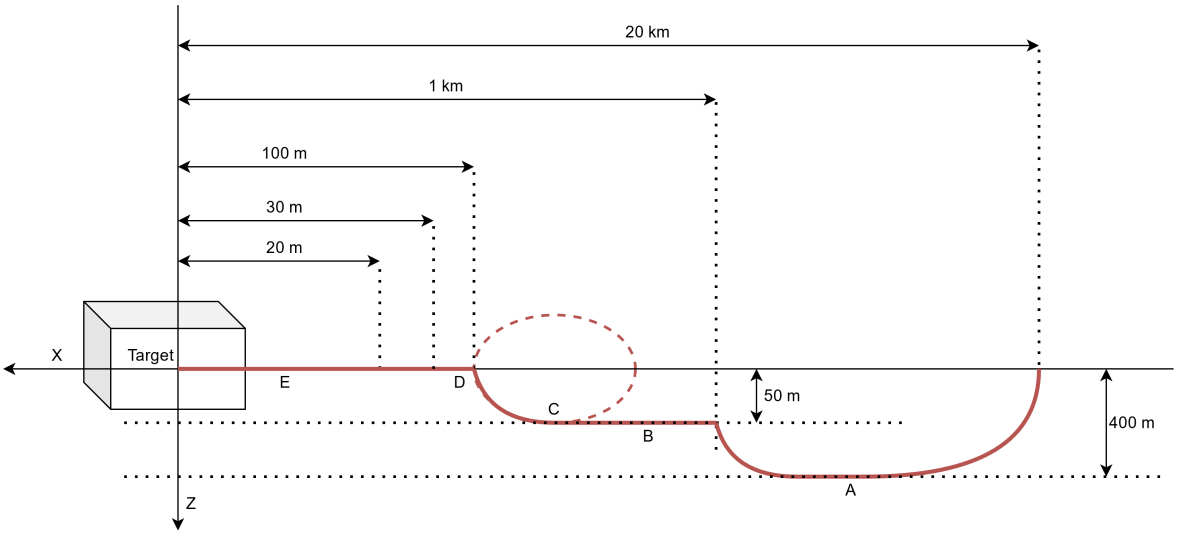


Figure 3.2: Typical approach profile of ADR mission, based on: [53]

terms of rotational rate of the target, relative velocity and relative 6-DOF pose. While Colomenarejo et al. also mention the determination of the centre of mass of the space debris, this is neglected for the present use case since it does not have implications on the fiducial marker system. After this characterisation, an approach strategy is used to approach the target to the desired minimal range at E. In the present use case, this range is assumed to be 1 metre. From 20 metres up to 1 metre, a fiducial marker system is assumed to be used for relative navigation.

The environment of the use case is broadly defined. In terms of illumination eclipse, partial sunlight, full sunlight, etc. are all considered in accordance with the findings in section 2.4. The main application area of the system is in LEO.

To further define the use case of the marker system, the end-to-end relative navigation system modes are defined that are relevant for the RENDER system. This is necessary for the derivation of requirements for the marker system. The modes are shown in Figure 3.3. Three modes are defined: spot mode, characterisation mode and track mode. Before the initialisation of any of these modes, the system might perform secondary functions such as the spotting of the spacecraft or line-of-sight and range measurements. These functions are considered out of scope for this thesis.

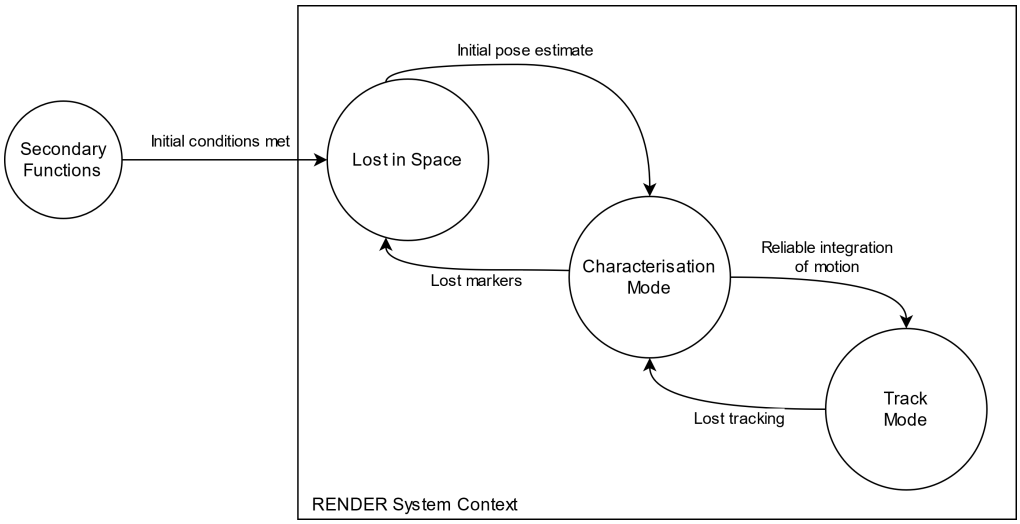


Figure 3.3: Relevant end-to-end relative navigation system modes for the RENDER system

Once certain initial conditions are met, such as a maximum relative range to the target and confirmation of target in FOV (the modes for this functionality are not relevant to the scope of RENDER), the system can be initialised in Lost in Space (LiS) mode. In LiS mode, the system attempts to localise the fiducials on the spacecraft and perform initial acquisition of the relative pose. Note that the spacecraft is not actually "lost in space", the relative navigation system has however no a priori information on the relative pose (in an end-to-end spacecraft this is possible, but it is assumed that this is not the case). The system remains in this mode until the markers are spotted and a reliable initial pose estimation is achieved (this may require the spacecraft to change relative position). Once an initial pose estimate is produced, the system will initialise characterisation mode in which subsequent solutions of the initial relative pose estimation problem are integrated to solve for relative velocity and rotation. Once these are established, the system can switch to track mode. In this mode, a navigation filter enables the estimation of the position of the fiducial markers and allows for faster, more accurate solutions of relative motion and pose.

It is assumed that the performance parameters in terms of accuracy and precision are those of the LiS pose estimation solution. This means that in terms of research boundary for this thesis, from the three defined modes this thesis is concerned with the LiS mode. An end-to-end system with all modes implemented includes a navigation filter, additional sensors such as an IMU etc. that influence the performance of the relative pose estimation. However, to characterise the performance of a fiducial marker system alone it is necessary to analyse the LiS performance.

Besides the end-to-end system concept, the context of the fiducial marker system is used in the translation of stakeholder requirements to system requirements. This context shows the system boundaries and is used in identifying driving factors for the system requirements of the fiducial markers, to develop STA-REQ-1 into system requirements. In Figure 3.4 these external drivers for the fiducial marker design are shown.

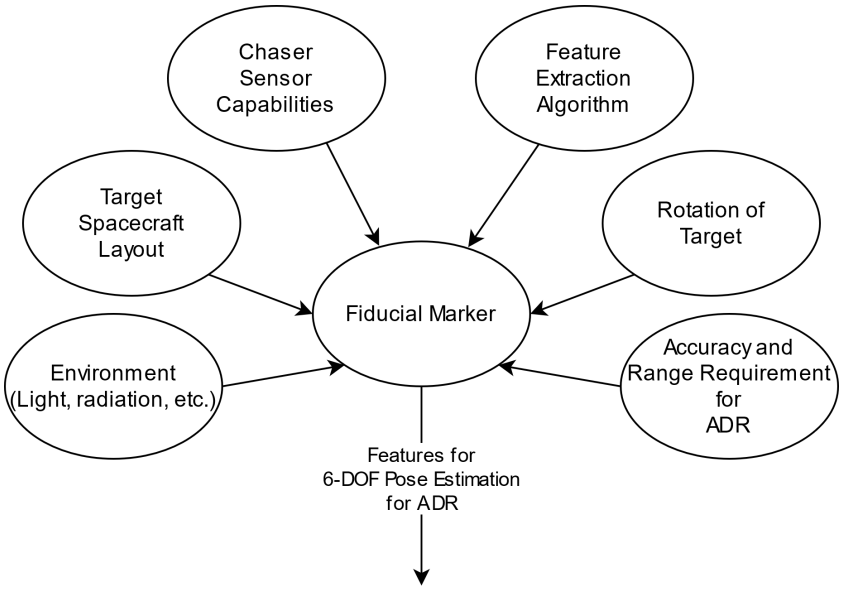


Figure 3.4: System context diagram for RENDER

3.6. System Requirements

With the stakeholder requirements defined, it is possible to translate these into system requirements for the fiducial marker system i.e. to go one level lower in the Vee model as depicted in Figure 3.1. The system context diagram and the stakeholder requirements are used as input to the requirement engineering process. To ensure completeness of the system requirements, a requirements discovery tree was used. This diagram is shown in Figure 3.5.

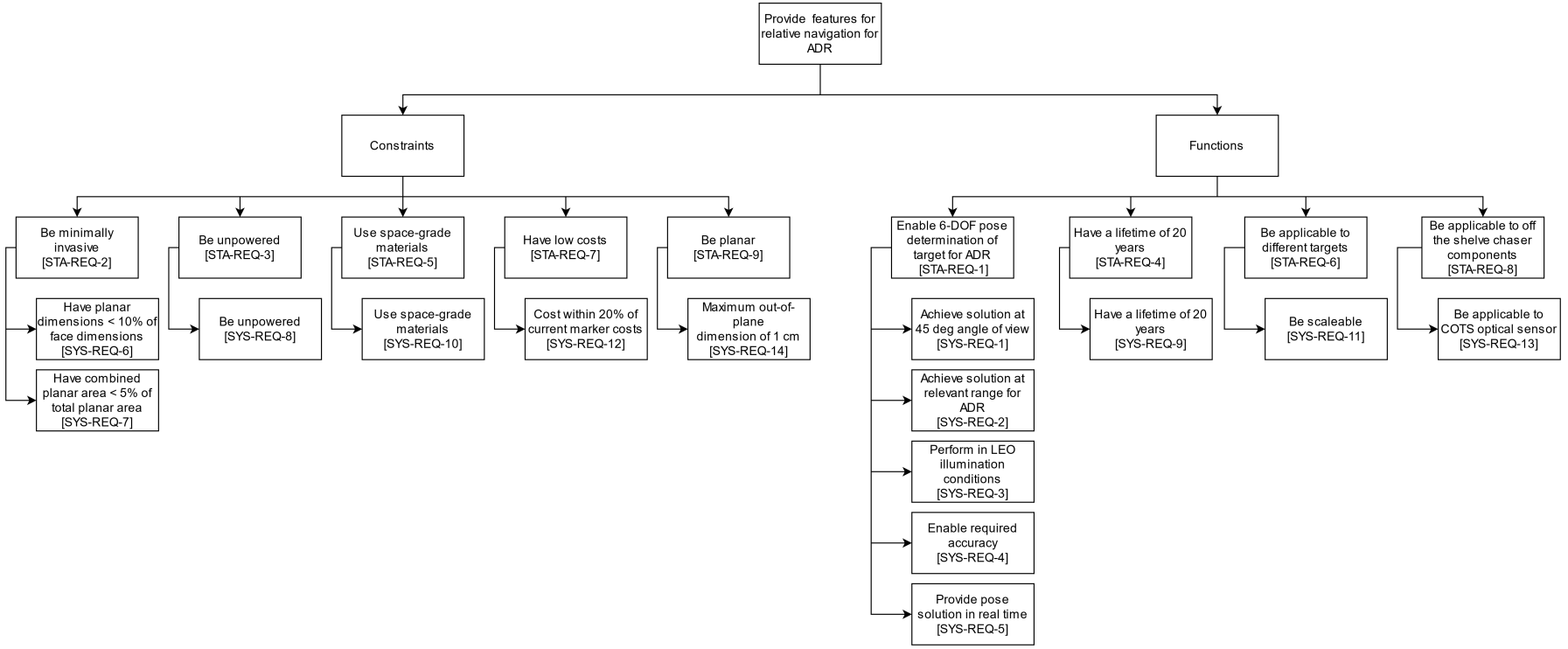


Figure 3.5: Requirements Discovery Tree

In order to define requirements and benchmark the performance of the fiducial marker, some external parameters need to be fixed. These parameters are the target spacecraft dimensions and chaser sensor capabilities. Since the research objective is not restricted to one type of spacecraft it is argued that the system needs to be inherently flexible in application. This is also captured in STA-REQ-6. However, some parameters need to be fixed in order to define a baseline of the system performance. Therefore, a historical reference system is chosen. This system is the Visual Based Sensor (VBS) by DTU. Choosing this system allows fixing the chaser sensor capabilities, shown in Table 3.1. These properties have been used as input to a Monte Carlo simulation for deriving accuracy and precision requirements for the end-to-end relative navigation system, as explained in section 3.7. It should be noted however that no assumptions are made on the illumination system, number of optical sensors or the spectral properties of the sensor system at this point.

Parameter	Value
Resolution Detector	752x580 pixels [54]
Pixel Size Detector	8.6 μm x 8.3 μm [55]
Focal Length	20.19 mm [56]

Table 3.1: System baseline parameters, based on VBS

The requirements have been constructed while ensuring adherence to the VALID criteria. A good requirement should be Verifiable, Achievable, Logical, Integral and Definitive [49]. To ensure that the requirements are verifiable, a verification method has been defined for each requirement. To ensure that the requirements are achievable, a baseline system has been defined thus ensuring achieve-ability by analogy. Using the DOT and engineering judgement, the requirements are validated to be logical and integral. Requirement traceability is established for all requirements by defining the parent stakeholder requirement and the completeness of the requirements has been validated using the system concept of operations. Finally, the requirements are definitively established by engineering judgement. Each requirement covers only one aspect of the system design or performance. The requirements are expressed in terms of needs, and not solutions are suggested or excluded. Finally, they are unambiguous and complete.

The following system requirements have been derived:

[SYS-REQ-1] The marker system shall provide the number of features required for 6-DOF pose estimation under a maximal angle of measurement of 45 degrees, with a tolerance of 5 degrees. *Rationale:* The goal angle of measurement to the marker system is based on a typical angle of acceptance of a retro-reflector [57], which has been validated as a fiducial marker for relative navigation in space. *Parent:* STA-REQ-1. *Criticality:* Medium. *Verification method*: Analysis, test by experiment.

[SYS-REQ-2] The marker system shall provide the number of features required for 6-DOF pose estimation at a range of 20 to 1 metres, with a tolerance of maximum range of 5 metres. *Rationale:* Range requirement is based on the use case analysis presented in section 3.5. *Parent:* STA-REQ-1. *Criticality:* Medium. *Verification method*: Test and analysis.

[SYS-REQ-3] The marker system shall provide features in all LEO illumination conditions. *Rationale:* According to [31], it should be assumed that it is not possible to plan for favourable illumination conditions in ADR missions. *Parent:* STA-REQ-1. *Criticality:* Medium. *Verification method*: Test and analysis.

[SYS-REQ-4] The accuracy, precision and outlier rate in lost-in-space mode of the end-to-end relative pose estimation shall be as shown in Figure 3.6 with a margin of 25%. *Parent:* STA-REQ-1. *Rationale:* See section 3.7. A large margin is justified by the *qualitative* nature of this requirement. *Criticality:* High. *Verification method*: Analysis.

[SYS-REQ-5] The marker system features shall be extracted in real-time. *Parent:* STA-REQ-1. *Rationale:* As the target may be spinning, it is necessary to perform relative pose estimation in real-time. Real-time is defined as fast enough to provide integration of motion with a target rotational rate of 5 deg/s, in accordance with the rotational rate requirement of the Envisat removal mission e-deorbit [31]. *Criticality:* Low. *Verification method*: Analysis and inspection.

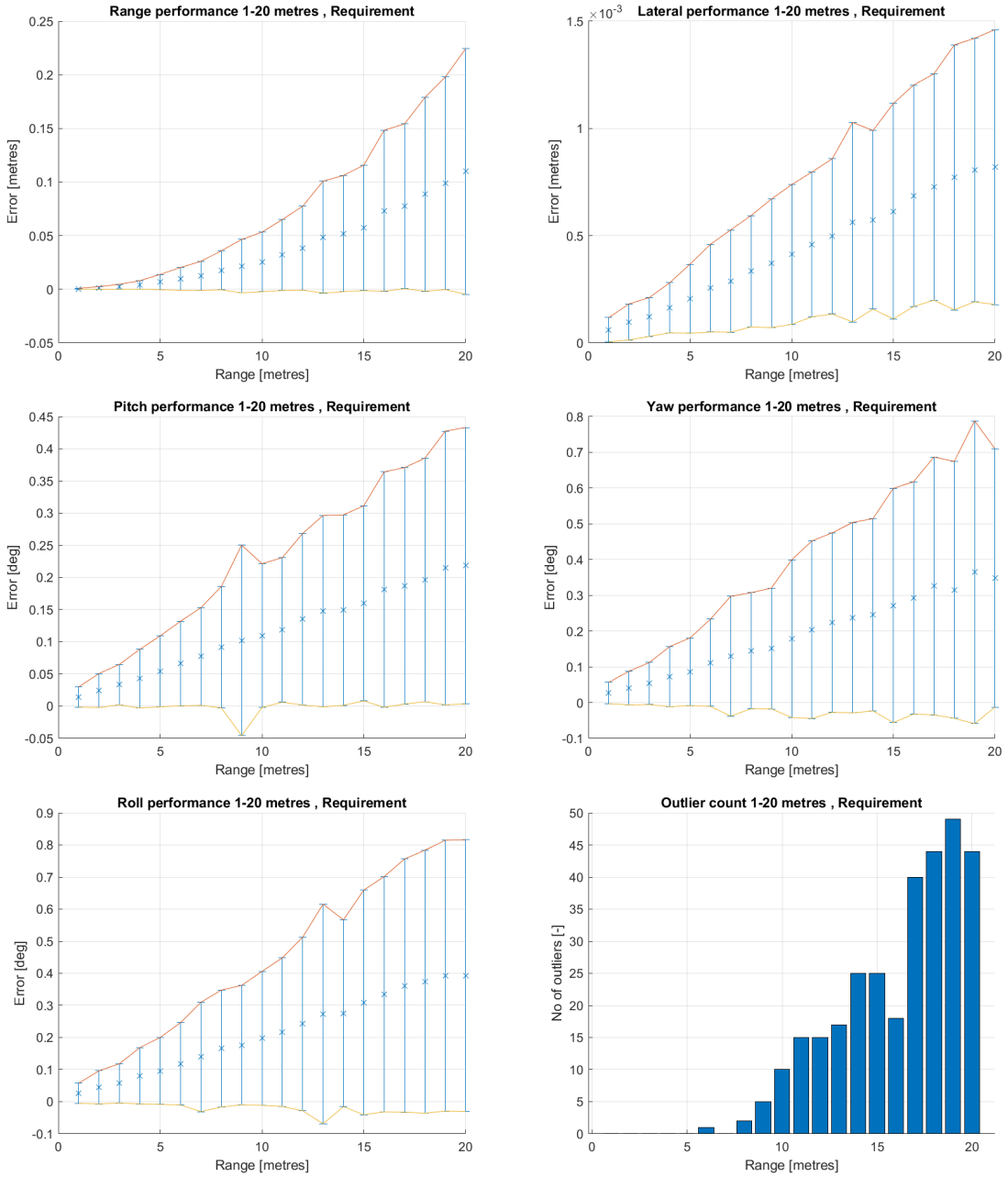


Figure 3.6: Accuracy and precision requirements as specified in [SYS-REQ-4]. n=1500 per range step of 1 metre.

[SYS-REQ-6] Planar dimensions of individual markers shall not exceed 10 % of both target face dimensions. *Rationale:* Individual markers should use a small relative amount of surface in accordance with STA-REQ-2. Furthermore, individual markers should be such that they can be flexibly placed on the surface of the spacecraft. *Parent:* STA-REQ-2. *Criticality:* Medium. *Verification method*: Inspection.

[SYS-REQ-7] The total combined planar area of the individual markers shall not exceed 5 % of the target face area. *Rationale:* The total area of the markers should use a small relative amount of surface in accordance with STA-REQ-2. *Parent:* STA-REQ-2. *Criticality:* Medium. *Verification method*: Inspection.

[SYS-REQ-8] The marker system shall be unpowered. *Rationale:* Same as STA-REQ-3. *Parent:* STA-REQ-3. *Criticality:* High. *Verification method*: Inspection.

[SYS-REQ-9] The marker system shall have a lifetime of 20 years. *Parent:* STA-REQ-4. *Rationale:* Same as STA-REQ-4. *Criticality:* High. *Verification method*: Analysis.

[SYS-REQ-10] The marker system shall use only space-grade materials. *Parent:* STA-REQ-5.*Rationale:* Same as STA-REQ-5. *Criticality:* Medium. *Verification method*: Inspection.

[SYS-REQ-11] The marker system shall be applicable to spacecraft faces of 1x1 metres up to 10x10 metres, with cost being at most being linearly dependant on system volume. *Parent:* STA-REQ-6. *Rationale:* To enable standardisation and in order to be applicable to a wide range of target spacecraft sizes, the marker system should be scalable in terms of size without requiring expensive redevelopment. Therefore, no elements should be used that are expensive to scale. *Criticality:* High. *Verification method*: Analysis and inspection.

[SYS-REQ-12] The marker system costs, including development costs and integration costs, shall be within an estimated 20% margin of current marker system costs. *Parent:* STA-REQ-7. *Rationale:* Directly derived from STA-REQ-7. Since a quantitative cost analysis is not feasible at this phase of development, requirement verification shall be done by qualitative analysis. For current marker systems, the Space Vision Marker System (SVMS) [47] is taken as benchmark. *Criticality:* Medium. *Verification method*: Analysis and expert review.

[SYS-REQ-13] The marker system shall be applicable to a COTS optical sensor system. *Parent:* STA-REQ-8. *Rationale:* Directly derived from STA-REQ-8. *Criticality:* High. *Verification method*: Inspection.

[SYS-REQ-14] The maximal out-of-plane dimension of the markers shall be 10 mm, with a tolerance of 5 mm. *Rationale:* The system should be planar in order to adhere to STA-REQ-9. *Parent:* STA-REQ-9. *Criticality:* High. *Verification method*: Inspection.

The capabilities described in [SYS-REQ-4], [SYS-REQ-9], [SYS-REQ-11] and [SYS-REQ-13] are the identified Key Performance Parameters (KPPs) for RENDER, since these are the functional requirements with the highest criticality.

3.7. Monte Carlo Simulation for Accuracy and Precision Requirements

All written code for the Monte Carlo simulation can be found in Appendix A.

To decompose [STA-REQ-1] into complete system requirements in accordance with the requirements discovery tree, it is necessary to qualitatively define precision and accuracy requirements for the pose estimation of the fiducial marker system. After all, if the system is to enable relative navigation for typical ADR missions, when can it be said that this customer need is fulfilled? To answer this question, a Monte Carlo simulation of the end-to-end system performance is made.

In order to qualitatively define what entails satisfactory accuracy and precision requirements, two assumptions are made. Firstly, it is assumed that a random uniform distribution of 8 feature points in a volume of 0.5x0.5x0.1 metre is an appropriate qualitative benchmark of the performance of a fiducial marker system. Essentially, this mimics the performance one would achieve if, for example, 8 uniquely identifiable retroreflectors were randomly placed on a spacecraft face (with reflectors at different protrusions). This number of feature points is redundant when considering the number of retroreflectors used in historical relative navigation systems [21]. Furthermore, it should be noted that this system would not be feasible under the stated requirements: the protrusion of the retroreflectors is not allowed under [SYS-REQ-14]

Secondly, it is reasonable to assume that the end-to-end system performance *under the fixed parameters of the simulation performed here* is only dependant on the geometry of the feature points used in the PnP solver and the relative pose of the optical sensor and target. In keeping the sensor parameters and PnP solver constant as well as assuming that for the purpose of this analysis all feature extractions are performed *flawlessly* (i.e. all features are always extracted and correctly identified) the geometry of the proposed fiducial marker system can be tested for qualitative performance with respect to using 12 distributed feature points. As stated above, the intrinsics of the optical sensor are set. These parameters are listed in Table 3.1. Using these parameters, a Monte Carlo simulation of a relative pose estimation between an optical sensor and a set of points is made using MATLAB.

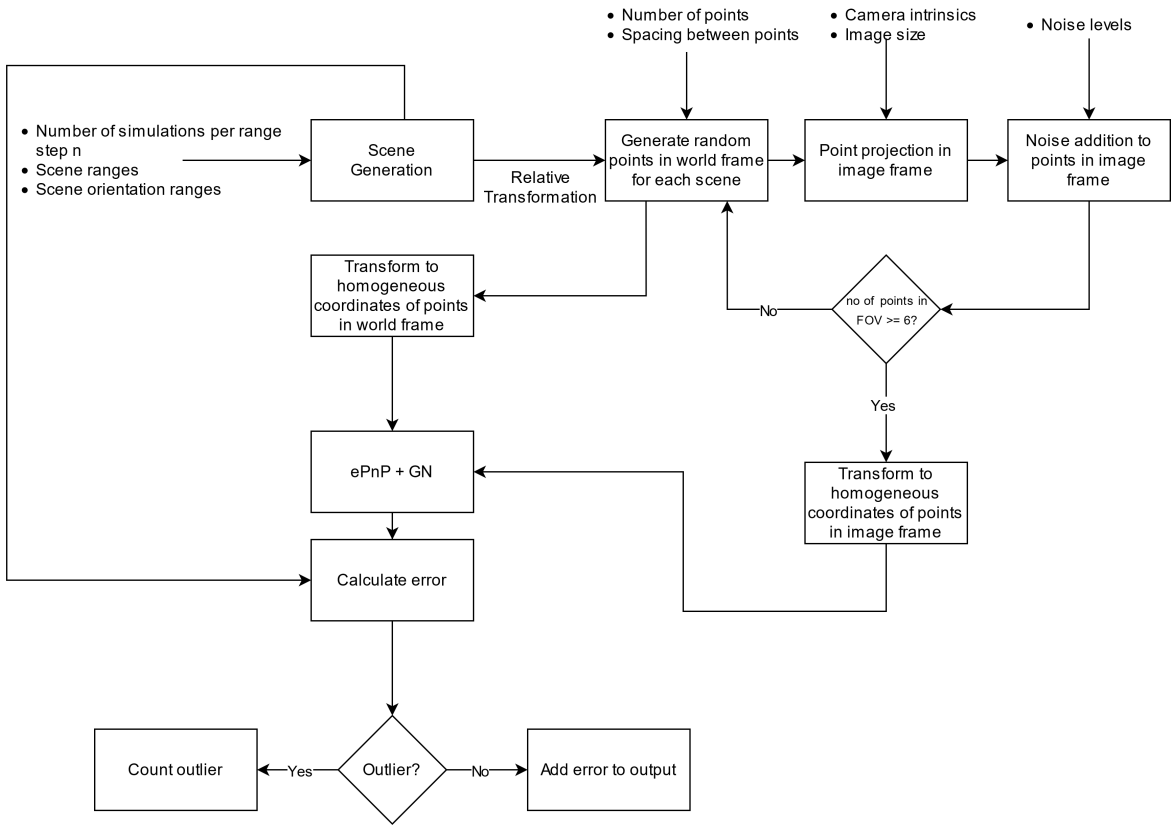


Figure 3.7: Schematic representation of Monte Carlo simulation for accuracy and precision requirement modelling.

A description of the used simulation setup is shown in Figure 3.7. The simulation is initialised by generating a range of "scenes" i.e. relative poses between the camera and artificial points. Within certain defined boundaries in terms of relative pose, an uniformly distributed random sample is taken from the infinite possible scenes within the operational envelope of the fiducial marker system at step increments of 1 metre. In terms of pitch and yaw, these boundaries are defined as -45 to 45 degrees in accordance with [SYS-REQ-1]. The number of simulations n per distance step is set which defines how many scenes are generated at that distance. The function block that produces the randomly generated points in the world frame is schematically shown in Figure 3.8.

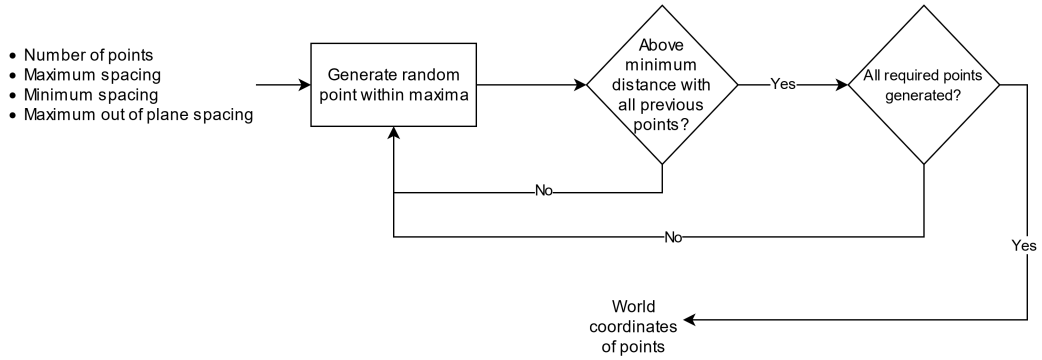


Figure 3.8: Schematic representation of point generation function of Monte Carlo simulation.

After these points are generated in the world frame, the points are projected onto the image frame using the MATLAB function worldToImage[1]. This function takes as input the relative transformation of the scene, the camera intrinsics and image size. The output is the coordinates of the artificial points on the image frame in pixel coordinates. A level of noise is added, modelled as a Gaussian shift in pixel coordinates. Calhoun et al. modelled this noise for a laser-illuminated retroreflector based navigation system in [58]. The authors assume that noise is a function of the incident power of the reflected laser beam, which scales by the range squared, and the spot diameter of the reflector which scales linearly with range for small subtended angles. Additionally, the authors assume a minimum and maximum centroiding error of 0.1 and 0.5 at minimum range and 100 metres, respectively. Then the centroiding noise can be modelled as follows:

$$noise = n_{min} + \frac{n_{max} - n_{min}}{R_{max}} * R_i^3 \tag{3.1}$$

Where n_{min} and n_{max} are the noise levels at the minimum and maximum range in pixels. R_i is the range in metres. While the authors use the modelled noise value to bound the centroiding error, it is used here as the standard deviation of the centroiding noise for the specific range. This is done since it preserves the range relation of the noise while producing a more conservative estimate. The noise is added to both dimensions of the feature point coordinates.

After the addition of noise to the feature points, a conditional is called that checks if the number of points in the FOV is equal to or larger than 6. This is necessary since the implementation of the PnP solver used requires at least this number of points. If the required points are not present, the point generation function is called again. If the required points are present, the feature point coordinates are transformed to homogeneous coordinates and together with the homogeneous world coordinates fed into the EPnP solver with Gauss-Newton optimisation. The output of this solver is compared to the scene pose, and the error is calculated. If the error is extremely large or impossible, such as a solution with a negative range, the solution is discarded and the outlier is added to an outlier count for that range. The ratio of outliers to solutions is added to the performance requirements of the system.

3.7.1. Convergence Study

To estimate the required number of runs per distance n required for convergence of the simulation, a convergence study is performed. The result of the convergence study is shown in Figure 3.9.

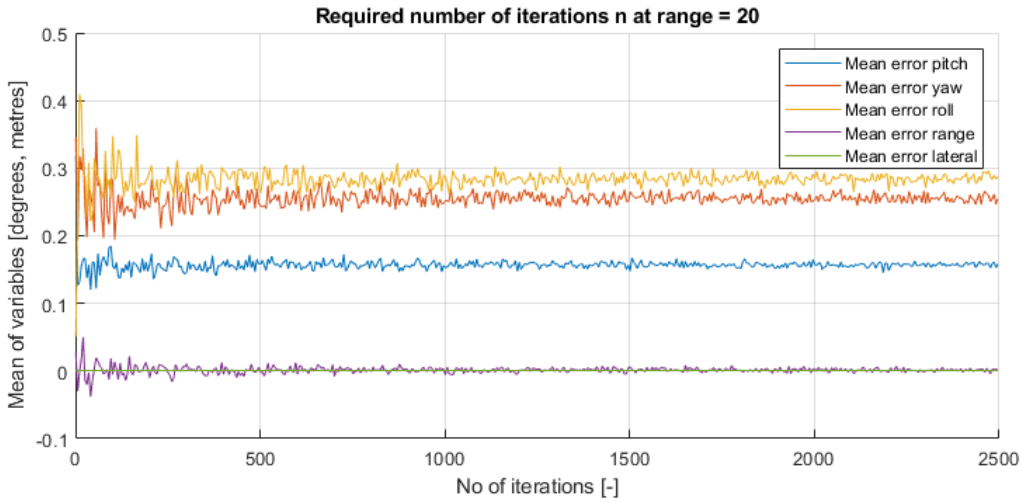


Figure 3.9: Convergence study for Monte Carlo simulation.

[1]Online Documentation: https://nl.mathworks.com/help/vision/ref/worldtoimage.html, accessed 23-4-2021.

This result was generated by running the Monte Carlo simulation at the maximum range of 20 metres for an increasing n from 1 to 2500. This distance is chosen since the performance of the system is the lowest at the largest range and thus the variance of the Monte Carlo simulation is the highest at this distance. The results for the error measures are recorded for each run with number of simulations n. The law of large numbers predicts that as $n \to \infty$, the solutions of the Monte Carlo simulation converge towards the real value of the errors that are to be found. However, only a stable solution to a 10th of a degree or so is required to interpret the system performance qualitatively. All errors converge quickly, at around $n = 1200$ the solutions are stable to within 0.1 degrees for rotations and centimetre level for translation. To be conservative and since the resources are available, a value of $n = 1500$ is chosen as the number of iterations for the determination of the requirements.

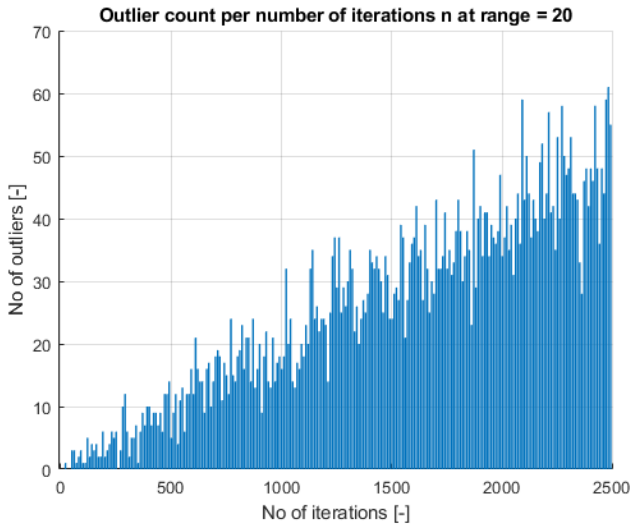


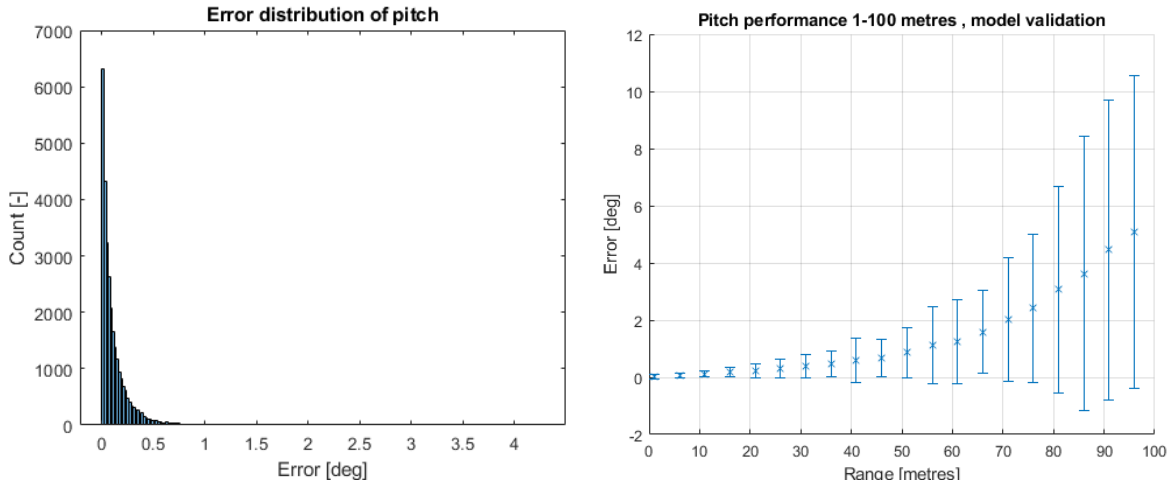
Figure 3.10: Outlier count per number of iterations n.

To validate the data used in this convergence study, the number of excluded outliers per run are analysed. This number should grow linearly with the number of iterations in order to prove that the stability shown in Figure 3.9 is not artificial due to a much lower measurement count. As shown in Figure 3.10, the number of outliers grows linearly with the increase of the number of simulations and is at 20 metres around 2% for the randomly generated points. This validates the found number of iterations.

3.7.2. Model Validation

The Monte Carlo simulation is validated by inspecting the distribution of the calculated pose errors. Shown in Figure 3.11a is this distribution for the pitch error. This distribution shows no outliers, as would be expected due to the filtering applied. Furthermore, the data shows only positive values and appears to be exponentially distributed with the highest chance of having an error that approaches zero. All other error measurements show the same behaviour, which is expected for this simulation.

The model is additionally validated by comparison to a similar model. Interestingly, the model of Calhoun et al. in [58] show an exponential increase in error at a larger range (up to 100 metres). Due to the exponential increase in noise as well as the exponentially reduced pixel area per square metre at larger ranges, such behaviour would be expected. However, in the requirement plots in Figure 3.6, the error appears to increase linearly with range. To validate if this is the case, the model is run again up to an extended range of 100 metres. In Figure 3.11b the result is shown, again for the pitch error measurement. The behaviour of calculated error is indeed exponential at a larger range. This result therefore in line with existing models, and validates the model presented here.



(a) Distribution of pitch error for requirements simulation.

(b) Pitch error for range of 1 to 100 metres.

Figure 3.11: Model validation for Monte Carlo simulation

3.8. Conclusion

A systematic system engineering approach is used in the design of the RENDER system. A need and mission statement were identified for the system and subsequently all stakeholders were identified. Flowing down from the identification of all stakeholders, the stakeholder requirements were identified.

In the concept of operations, the system boundary for RENDER was identified. The system shall be used for the last 20 metres of a rendezvous of an ADR mission. The main application is in LEO. Furthermore, the performance of the system shall be measured in the LiS mode to not have any influence of e.g. the navigation filter on the fiducial marker system performance measurement.

The system requirements were derived using a DOT and with several set parameters based on a reference mission. While these parameters were set for the requirements definition, the system is explicitly not limited to application in a system with these parameters. The system requirements were validated using the VALID criteria.

For the qualitative accuracy, precision and outlier requirement, a Monte Carlo simulation was made. The number of required simulations was found using a convergence study. This simulation was validated using the distribution of the found data and comparison to a similar existing model.

With the system requirements defined, the next step in the system engineering process is moving to the design synthesis side of the Vee model.

4

System Concept Selection

With the establishment of the concept of operations, stakeholder and system requirements, the system concept needs to be selected. The design options for the RENDER system concept at the highest level are based on a taxonomy of fiducial markers for relative navigation in space. To perform a quantified trade-off for the RENDER system concept selection, an Analytic Hierarchy Process (AHP) has been used.

The design options for the system concept selection are presented in section 4.1. Next, in section 4.2, tradeoff criteria are listed and using the AHP the design options are analysed and a system concept is selected. In section 4.3 the performed tradeoff is validated by a sensitivity analysis.

4.1. Design Options for System Concept Selection

In the literature review that preceded this thesis, a high-level taxonomy of fiducial markers for passive relative navigation in space has been established [21]. This taxonomy is refined here to provide a basis for a trade-off of the design options for the system concept for RENDER.

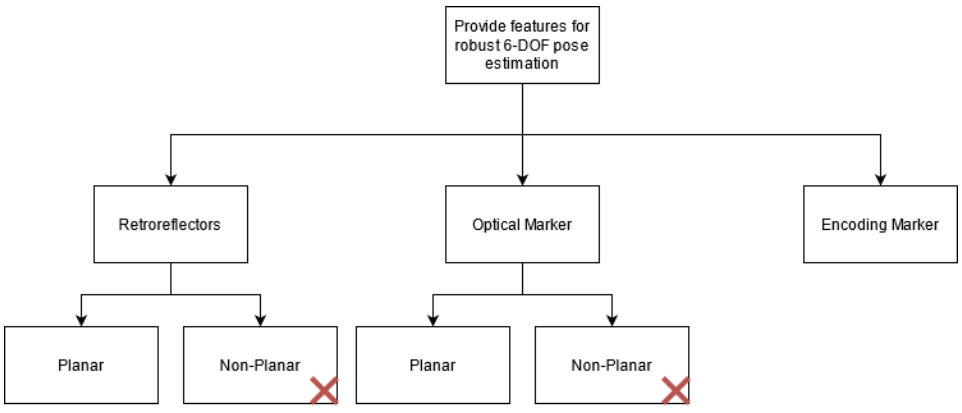
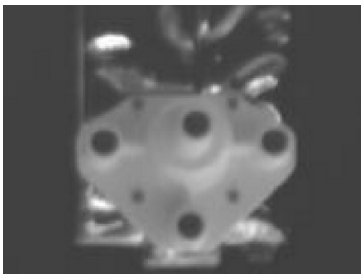
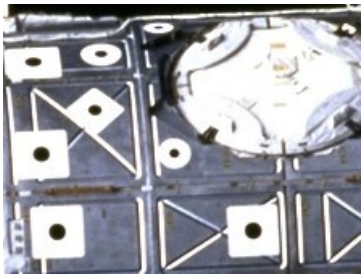


Figure 4.1: Design Option Tree (DOT) for the system concept selection of RENDER

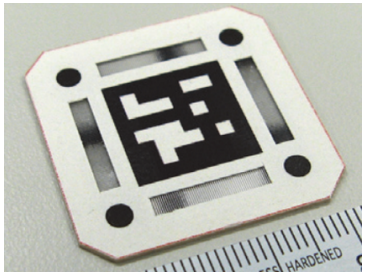
First, a tradeoff is performed for the highest level of the design of the RENDER fiducial marker system: the system concept selection. The 3 defined fiducial marker types are shown in the Design Option Tree (DOT) shown in Figure 4.1. The design options are based on an extensive literature survey of fiducial marker systems [21].



(a) The short range retroreflector Advanced Video Guidance Sensor (AVGS) marker system for the Orbital Express mission. Reprinted from [59]. Image by NASA.



(b) Space Vision System (SVS) optical fiducials on the ISS. Image by NASA.



(c) Encoding marker Lentimark. Reprinted from [60], with permission. ©2014 IEEE

Figure 4.2: Design options for the RENDER concept selection. (a) shows the retroreflector, (b) shows the optical fiducial, (c) shows the encoding marker.

From left to right in Figure 4.2, the three design options for the system concept are:

1. Retroreflectors - The retroreflector (shown in Figure 4.2a) has been used extensively in relative navigation systems for spaceflight [24, 47, 61–63]. The system consists of several retroreflectors (usually corner reflectors). These retroreflectors can be applied distributed individually or in a target configuration. These markers require contextual identification for pose estimation and are applicable to very large range (300+ metres) pose estimations [64]. Retroreflectors provide an added benefit from a systems perspective of facilitating rotation estimation of space debris targets from ground.

2. Optical marker - Optical markers (shown in Figure 4.2b) are defined here as markers that do not have retroreflective properties nor the ability to encode an additional layer of pose information in the signal provided to the chaser sensor. Note that these markers, although named "optical" here, do explicitly not use any optical elements. The optical marker has been used extensively in space, in both robotic applications and relative navigation [22, 23, 47, 65, 66]. In addition, these markers have been extensively used in robotic applications on Earth and many software libraries that generate and recognise these markers exist [3–6]. Even for "terrestrial" optical markers, space applications for rendezvous have been suggested [7]. These markers may be uniquely identifiable if an encoding bit pattern is present. Usually the corners of the fiducial are used for providing features to the PnP solver. These markers may also be of simpler geometry and may require contextual identification, as is the case with the example shown in Figure 4.2b.

3. Encoding marker - The encoding marker (shown in Figure 4.2c) has, to the knowledge of this author, not been used in space applications nor has it been used outside of experimental settings in terrestrial applications. As first envisioned by Bruckstein et al. in [67], these markers directly encode pose information in terms of greyscale, colour or temporal signals. This definition is extended here to include markers that encode pose information using a "spatial" signal e.g. a feature that has a variable position on the image plane of the chaser sensor dependant on the relative pose. An example of an encoding marker is the Lentimark [60]. This particular marker uses lenticular lenses to encode relative pitch and yaw information in a grey stripe on perpendicular strips.

Shown in Figure 4.1, two subcategories are defined for the retroreflector and the optical marker design option: planar and non-planar. It has been shown in section 2.7 that planar features suffer from ambiguities in the pose solution causing the PnP solver to find the wrong solution. This problem may be solved by adding an out-of-plane feature to the fiducial which solves the ambiguity. However, any out of plane feature violates [SYS-REQ-15] and these design options are therefore dismissed.

4.2. Analysis of Design Options and Concept Selection

Saaty published a structured approach to decision making [68, 69] called the Analytic Hierarchy Process (AHP). This process shall be used here to decide on the most suitable system concept for RENDER. A relative comparison is made to determine the best fiducial marker for debris removal with respect to the derived system requirements.

In accordance with the AHP method as proposed by Saaty, the problem is structured as a hierarchy. The top-level or goal of this tradeoff is the "appropriate fiducial concept". The second level consists of the tradeoff criteria. The third level in this hierarchy consists of the three marker concept options: retroreflective, optical and encoding.

The criteria for the AHP are based on the KPPs as they are defined in section 3.6. One criterion is added to these KPPs: system complexity. These criteria are as follows:

1. Accuracy of pose solution
2. Precision of pose solution
3. Lifetime of fiducial
4. Scaleability of fiducial
5. Applicability to COTS sensor
6. System complexity

With the definition of the criteria and the judging scale, a pairwise comparison matrix for the first level in the hierarchy can be populated. The criteria are judged using the scale as defined by Saaty, presented in Table 4.1.

Intensity of importance on an absolute scale	Definition	Explanation
1	Equal importance	Two activities contribute equally to the objective
3	Moderate importance of one over another	Experience and judgement strongly favour one activity over another
5	Essential or strong importance	Experience and judgement strongly favour one activity over another
7	Very strong importance	An activity is strongly favoured and its dominance demonstrated in practice
9	Extreme importance	The evidence favoring one activity over another is of the highest possible order of affirmation
2,4,6,8	Intermediate values between the two adjacent judgments	When compromise is needed
Reciprocals	If activity i has one of the numbers assigned to it when compared with activity j, then j has the reciprocal value when compared with i	
Rationals	Ratios arising from the scale	If consistency were to be forced by obtaining n numerical values to span the matrix

Table 4.1: Scoring scale used for the AHP. Reprinted from [68], with permission. Copyright (1990) Elsevier B.V.

The accuracy of the pose solution is considered as the comparison metric of the highest importance, followed by the precision of the pose solution. Accuracy is telling for the overall performance of the fiducial, where precision is telling for the stability of the pose solution (i.e. if many pose ambiguities occur). System complexity is considered the third most important metric, as this metric captures development risks, TRL, and system costs. Fourth is the lifetime of the fiducial, as this is critical to the effectiveness of the fiducial, i.e. essential for the fulfilment of the system objective, but not as essential as the previously mentioned metrics. The fifth most important is the scalability of the fiducial, which would greatly increase the application space of the system.

	1	2	3	4	5	6	Priority vector
1	1	2	4	4	5	3	0.373
2	$\frac{1}{2}$	1	3	3	4	2	0.241
3	$\frac{1}{4}$	$\frac{1}{3}$	1	2	3	$\frac{1}{2}$	0.108
4	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{2}$	1	2	$\frac{1}{2}$	0.078
5	$\frac{1}{5}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{2}$	1	$\frac{1}{3}$	0.050
6	$\frac{1}{3}$	$\frac{1}{2}$	2	2	3	1	0.149

λ_{max} = 6.1555 CI = 0.0311 CR = 0.0249

Table 4.2: Pairwise comparison matrix for the second level of the AHP.

Finally, applicability to COTS sensors has the lowest importance, since this KPP greatly increases application space but could be mitigated by applying a custom sensor.

For the comparison matrix shown in Table 4.2, the principal eigenvalue is computed using MATLAB. The associated eigenvector is computed as well, which when normalised gives the priority vector indicating the found relative worth of the individual criteria. In an ideal situation, i.e. complete consistency, all rows are linearly dependent on each other. This is not the case for the populated comparison matrix here, causing the need to check if consistency is within acceptable margins as defined by Saaty in [68]. This is done by first calculating the consistency index (CI):

$$CI = \frac{\lambda_{max} - n}{n - 1} \tag{4.1}$$

Where n is the number of criteria. Using this consistency index, the consistency ratio can be determined using a random index (RI) which has been calculated using a large number of matrices with random judgement to be 1.25 for n=6 [70]. According to [68, 70], if the consistency ratio (CR) is below 10%, the matrix consistency is acceptable.

$$CR = \frac{CI}{RI} \tag{4.2}$$

The calculated values are shown in Table 4.2, showing that the matrix indeed passes the consistency criteria.

Next, in the lowest level in the hierarchy, the 3 design options are scored with respect to another using all defined criteria. This yields six 3x3 matrices (shown in Table 4.3), to which the same consistency checks are applied as above. For these matrices, option 1 is taken to be the retroreflective option, option 2 the optical option and option 3 the encoding option.

In addition to the priority vector, an "Ideal Mode" normalised priority vector is calculated. Belton and Gear found that rank reversal could occur if a similar design option to one of the existing is introduced [71]. The authors proposed an adaption of the original AHP later accepted by Saaty in [72] called the "Ideal Mode"

Accuracy of pose solution

	1	2	3	Priority vector	Ideal mode Pv
1	1	2	$\frac{1}{2}$	0.297	0.550
2	$\frac{1}{2}$	1	$\frac{1}{3}$	0.163	0.302
3	2	3	1	0.540	1.000

λ_{max} = 3.0092 CI = 0.0046 CR = 0.0088

Precision of pose solution

	1	2	3	Priority vector	Ideal mode Pv
1	1	$\frac{1}{2}$	$\frac{1}{6}$	0.111	0.166
2	2	1	$\frac{1}{3}$	0.222	0.333
3	6	3	1	0.667	1.000

λ_{max} = 3 CI = 0 CR = 0

Lifetime of fiducial

	1	2	3	Priority vector	Ideal mode Pv
1	1	4	5	0.683	1.000
2	$\frac{1}{4}$	1	2	0.200	0.293
3	$\frac{1}{5}$	$\frac{1}{2}$	1	0.117	0.171

λ_{max} = 3.0246 CI = 0.123 CR = 0.0236

Scaleability of fiducial

	1	2	3	Priority vector	Ideal mode Pv
1	1	$\frac{1}{2}$	3	0.292	0.475
2	2	1	7	0.615	1.000
3	$\frac{1}{3}$	$\frac{1}{7}$	1	0.093	0.151

λ_{max} = 3.0026 CI = 0.0013 CR = 0.0025

Applicability to COTS sensor

	1	2	3	Priority vector	Ideal mode Pv
1	1	5	5	0.714	1.000
2	$\frac{1}{5}$	1	1	0.143	0.200
3	$\frac{1}{5}$	1	1	0.143	0.200

λ_{max} = 3 CI = 0 CR = 0

System complexity

	1	2	3	Priority vector	Ideal mode Pv
1	1	$\frac{1}{5}$	2	0.167	0.226
2	5	1	7	0.740	1.000
3	$\frac{1}{2}$	$\frac{1}{7}$	1	0.094	0.127

λ_{max} = 3.0142 CI = 0.0071 CR = 0.0136

Table 4.3: Pairwise comparison matrices between the three design options in the AHP.

AHP [73]. In this mode, for each pairwise comparison in between the three design options, each value of the design options is divided by the highest score for that specific criterion, as shown in Equation 4.3.

$$Ia_{ij} = \frac{a_{ij}}{max(a_{i1}, ..., a_{in})} \tag{4.3}$$

Where a_{ij} is the measure of performance for criterion j for design option i. n is the number of design options.

The first scoring is performed on the accuracy of the pose solution criterion. The accuracy of feature extraction is the same for the optical and encoding marker, as both rely on corner, edge or blob detection using computer vision techniques. Both markers perform these operations on an image of a marker that has *intrinsically* no special reflective properties. The retroreflector has inherently better reflective properties, returning a spread out point as signal. The level of spread of the point, in theory, would allow subpixel feature detection to a higher accuracy than the two alternatives, where the level of signal spread is not controllable. However, the accuracy of the pose solution (mainly in frontal observations) is for both the retroreflective option and the optical option limited by the planarity requirement [74]. The encoding marker has the ability to mitigate planarity by encoding additional projection information in its signal, thus making it score higher than the two alternatives.

The second criterion to be scored is the precision of the pose solution. Similar to the accuracy requirement, the encoding marker has the ability to mitigate pose ambiguities by encoding additional information while the other two alternatives will encounter pose ambiguity (and thus loss of precision) due to the planarity requirement. The difference in precision with respect to the encoding marker is estimated to be slightly more than the theoretical difference in accuracy due to the pose ambiguity mitigation, which mitigates the finding of local minima in the error function of the pose estimation algorithm. In addition, due to the need for contextual identification in the retroreflective option (which can be mitigated by including a bit pattern in either the encoding or optical option) a risk of wrongful identification is present in the retroreflective option. This reduces the precision of the pose solution.

Third is the lifetime of the fiducial. Note that while in theory the lifetime of all fiducials could be the same if material properties, mechanisms, etc. are similarly qualified, this metric is meant to capture the *risk* associated with the achievement of this lifetime. In particular, retroreflectors have extensive space heritage and need the optical elements (glass, mirrors) used in these components can be assumed to have excellent degrading properties. However, any necessary coatings/paints required for optical fiducials need to be qualified for the space environment and current materials (especially white paints) show significant degradation under UV and atomic oxygen even at lifetimes much shorter than the required 20 years [75, 76]. Similar to optical fiducials, encoding fiducials are likely to require contrasting paints. Furthermore, the encoding elements present on these markers may require some additional qualification and in case of failure increase the risk to the lifetime of the fiducial. Therefore, these markers score slightly less on the lifetime criterion with respect to the optical marker.

Next, the scaleability of the markers is scored. This term in itself needs clarification and is defined here as it is defined in [SYS-REQ-11]: cost shall be linearly dependant on system volume. Any system that uses optical elements such as mirrors or lenses may be hard to scale in theory, but arguably the commonality and heritage of retroreflectors mitigates this factor somewhat. Optical markers are easily scaled and do not require extensive redesign or qualification. By nature, encoding markers are much harder to scale than the two alternatives. Due to the at this point undefined encoding element, which may or may not use optical elements but definitely has no heritage, such a marker may require extensive redesign and re-qualification after scaling.

The applicability to a COTS sensor is lower for the optical and encoding marker, due to the need for a larger detail in measurements. Due to the absence of an encoding bit pattern (which may be present in both the optical marker and encoding marker), only a spot has to be imaged. Spot identification is derived from the contextual position of the spots relative to another. Therefore, the required level of detail from the sensor is lower for the retroreflective marker. It is assumed that the requirements imposed on the sensor by both the optical and encoding marker are the same.

System complexity is the lowest (i.e. best score) for the simple optical marker. While it can be argued that the retroreflective option has relatively high complexity, due to the heritage of retroreflectors and the high TRL of these components, the system is scored better than the encoding marker. The encoding marker has no heritage, in fact, no space application concept even exists currently. Furthermore, these systems as they currently exist are highly complex and utilise usually optical elements. Therefore, this system concept scores lowest.

The third and final step in the AHP process is to establish the global priorities of the design options. Since the ideal mode AHP is applied, the individual scores are normalised with respect to the total score to give an overall percentage. As shown in Table 4.4, both the retroreflector and optical marker have very similar overall scores. The encoding marker achieves the highest score but also has the highest number of "lost" criteria. This is an indication that while this design concept, in theory, is the most suitable for ADR, current implementations of these markers score poorly on lifetime, scalability and system complexity. The requirements posed for these KPPs may even be *show stoppers*. It is therefore necessary to further analyse the design for this system concept at a lower level and innovate on currently available encoding marker systems. This shall be done in the next chapter.

	Accuracy	Precision	Lifetime	Scale-ability	COTS sen-sor	Complexity	Score	Norm. Score
Weight Factor	0.373	0.241	0.108	0.078	0.050	0.149		
Retroreflector	0.550	0.166	1.000	0.475	1.000	0.266	0.474	29.46%
Optical	0.302	0.333	0.293	1.000	0.200	1.000	0.461	28.69%
Encoding	1.000	1.000	0.171	0.151	0.200	0.127	0.673	41.85%

Table 4.4: Final trade off table showing global priorities of the AHP.

4.3. Tradeoff Sensitivity Analysis

Before additional analysis is performed on the chosen system concept, the robustness of the decision made with the AHP is tested. This decision-making tool, even when used in its ideal mode form, can reach wrong conclusions under certain circumstances [77].

The sensitivity analysis performed here is based on a two-prone approach: first, the sensitivity of the tradeoff with respect to a change in relative weight factors is calculated. Next, the sensitivity with respect to a change in relative priorities of the design options is calculated. The method used here is based on the work by Triantaphyllou and Sánchez in [78].

The sensitivity with respect to a change in relative weight factors is calculated using the equation presented in Equation 4.4 [78]. This equation is used to calculate *the most critical criterion* i.e. the criterion which requires the smallest change to incite a rank reversal.

$$\delta_{k,i,j} = \frac{P_j - P_i}{a_{jk} - a_{ik}} \text{for any } 1 \leq \text{i} < \text{j} \leq \text{M and } 1 \leq \text{k} \leq \text{N} \tag{4.4}$$

$$\delta'_{k,i,j} = \frac{\delta_{k,i,j}}{W_k} \text{for any } 1 \leq i < j \leq M \text{ and } 1 \leq k \leq N \tag{4.5}$$

Where $\delta_{k,i,j}$ is the value of the minimum change in weight factor W_k for a certain criterion k that causes a rank reversal between option i and option j. P_i is the preference for design option i. M is the number of alternatives and N is the number of criteria. $\delta'_{k,i,j}$ is the change in relative terms to the current weight factor. An additional condition is posed on the absolute value of minimum change in weight factor to ensure feasibility of the rank reversal:

$$\delta_{k,i,j} \leq W_k \tag{4.6}$$

The result of these calculations is shown in Table 4.5. As expected, the optical-retroreflective comparison is quite sensitive to a change in criterion weight factor. These design options were very close together in the overall score. In particular, the system complexity only needs a 6.72% change in weight factor in order to cause a rank reversal. This is the so-called per cent any critical criterion. However, this fact does not cause the outcome of the AHP to change. In fact, only a very significant (above 50% in all cases) change in weight factor for the accuracy and precision criteria causes rank reversal for any of the two comparison pairs that include the winning encoding marker option. The per cent top critical criterion, i.e. the most critical criterion for the top-ranking design option is system accuracy with 50.56%.

Pair of alternatives	Criterion					
	1	2	3	4	5	6
1-2	8.38%	-19.33%	10.15%	-18.93%	19.39%	-6.72%
1-3	73.82%	61.68%	n/a	n/a	n/a	n/a
2-3	-50.56%	-81.89%	n/a	n/a	n/a	n/a

Table 4.5: Sensitivity analysis of criteria for AHP

Next, a sensitivity analysis is performed to determine the most critical measure of performance. Similar to the sensitivity of the criteria, the minimum change of the current value of the measure of performance a_{ij} is defined as [78]:

$$\delta^{a'}_{k,i,j} = \frac{P_i - P_k}{P_i - P_k + W_j(a_{kj} - a_{ij} + 1)} \frac{1}{a_{ij}} \text{for any } 1 \leq i, k \leq M, \text{ and } 1 \leq j \leq N \tag{4.7}$$

Where, similar to Equation 4.6:

$$\delta^{a'}_{k,i,j} \leq 1 \tag{4.8}$$

Using these equations, the following sensitivity analysis can be made on the the measures of performance:

Pair of alternatives	Criterion					
	1	2	3	4	5	6
1-2	4.89%	16.13%	16.69%	12.89%	43.64%	12.63%
1-3	-54.03%	n/a	n/a	n/a	n/a	n/a
2-1	-5.61%	-12.06%	-14.99%	-26.48%	-47.08%	-29.97%
2-3	-86.93%	n/a	n/a	n/a	n/a	n/a
3-1	37.66%	75.55%	n/a	n/a	n/a	n/a
3-2	53.90%	62.14%	n/a	n/a	n/a	n/a

Table 4.6: Sensitivity analysis of measure of performance for AHP

The per cent any critical measure of performance is the accuracy of the retroreflective option with respect to the optical option, with a relatively low required percentage of change of 4.89%. This is again expected

since both design options were so closely scored. Per cent top measure of performance is the accuracy of the encoding marker with respect to the retroreflective marker, with a drastic difference of 37.66%. It is concluded that the tradeoff passes the sensitivity analysis since the minimum change of criteria weight factor as well as the minimum change of measure of performance to incite a rank reversal with respect to the chosen design concept (the encoding marker) is sufficiently large in all cases.

4.4. Conclusion

Three design options were identified for the system concept selection for the RENDER fiducial marker system, based on a taxonomy of fiducial markers for vision-based relative navigation found after an extensive literature study that preceded this thesis. These design options consist of a retroreflective concept, an optical marker concept and an encoding marker concept.

These design options were analysed using an AHP. Six criteria were identified for the tradeoff. These criteria were based on the KPPs for the system design. The encoding design options scored the best in the attainable accuracy and precision. Since these two criteria carry the highest weight factor according to the AHP, the encoding option was selected as design concept. The AHP was consequently validated using a sensitivity analysis. It was concluded that the required chance in criteria weight factor as well as the required chance in measure of performance to incite a rank reversal for the selected design option was satisfactory and thus the tradeoff outcome was accepted.

While the encoding design option scored the highest in the AHP, from the tradeoff analysis it has also become apparent that in terms of lifetime, scalability and complexity the currently existing encoding markers score poorly. It is concluded that the requirements for these characteristics imposed on the fiducial system may even be *show stoppers*. Therefore, in order to make the encoding marker concept feasible for the system design, a new type of encoding element is required that solves these issues.

5

A Novel Planar Encoding Fiducial for High Precision Pose Estimation: The Mantis Marker

Fiducial markers provide a distinct signal for 6 degrees of freedom pose estimation of target objects, generally using a monocular camera. While there are numerous planar markers, these markers lose accuracy in frontal observations and suffer from pose ambiguity. Furthermore, range accuracy is in many cases problematic. Some solutions in the form of encoding markers exist, but these markers are usually complex and hard to manufacture. In addition, current encoding markers provide retroactive correction of the pose solution, potentially reducing the accuracy of the solution. In order to solve these issues and in line with the requirement analysis, a new type of encoding fiducial is required. In this chapter, a novel planar encoding marker is presented, called the Mantis Marker. This marker is 3-D printable and projects on the image plane a virtual point that is located behind the plane of the fiducial in the world coordinate frame. Mathematically, this is equivalent to using a 3-dimensional fiducial with a protrusion. This allows a direct implementation in existing pose estimation algorithms, thereby greatly improving the performance of planar fiducials.

The state of the art for encoding elements is covered firstly in section 5.1. Next, the iterative design approach used in designing the Mantis Marker fiducial is covered in section 5.2. The actual design of both the encoding element as well as the complete fiducial system is explained in section 5.3. The Mantis Marker requires both feature extraction software as well as pose estimation software to function. These elements are described in section 5.4 and section 5.5 respectively. Finally, a parametric FOV analysis of the fiducial marker system is presented in section 5.6.

5.1. State of the Art for Encoding Elements

To provide a basis for the encoding element design, the driving subsystem for the feasibility of the encoding system concept, a survey of existing elements is presented.

Research into encoding fiducials has been limited. To the knowledge of the author of this thesis, Bruckstein et al. provided the first definition for the encoding fiducial in 2000: *"objects whose images directly encode, in easily identifiable grey-level/colour or temporal patterns, the pose of their viewer."* [67]. As stated in chapter 4, this definition is extended to include objects whose image directly encode the pose of the viewer in spatial patterns. E.g. an encoding element that has a variable position on the image plane of the chaser sensor dependant on relative pose. Paraphrasing Bruckstein et al. in [79], encoding fiducials are defined here as:

Fiducial markers whose appearance beyond perspective effects changes with the relative pose of the viewer with respect to the marker.

A number of earlier patents have proposed what essentially are encoding fiducials [67]. In a 1979 patent, Bergkvist proposed a device that can be used to determine whether or not the viewer is on a plane defined by the device and that tells the viewer how to move relative to the device to reach the defined plane [80]. The device utilised so-called Moiré patterns, an example of which is shown in Figure 5.1. These patterns consist of two (or more) sets of lines placed on top of each other with different line spacing or different relative angles. When placed in specific settings, movement of the viewer can change the appearance of these patterns.

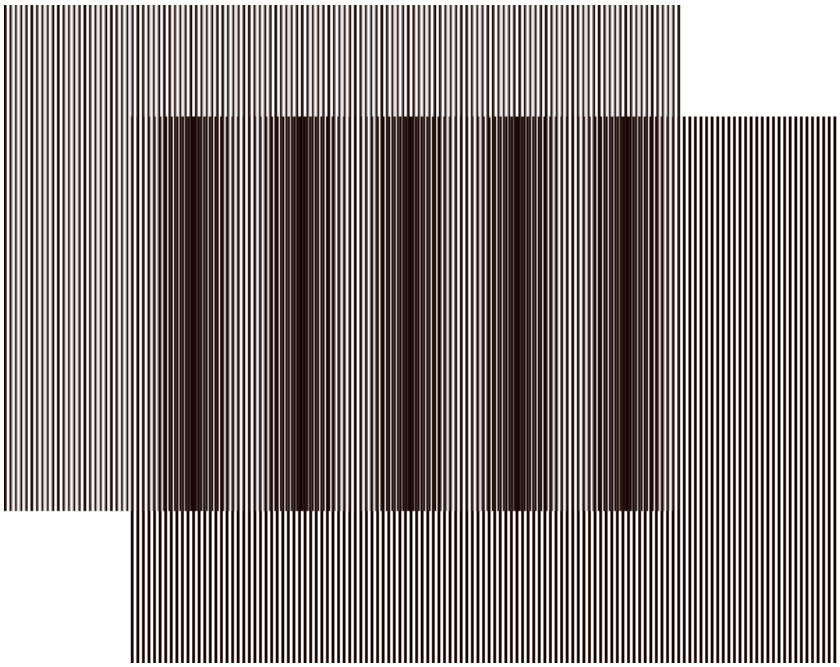


Figure 5.1: Moiré pattern. Image by Anton (rp) 2004. CC BY-SA 3.0

In a 1984 patent by Kunkel, a fiducial that utilises the shadow caused by an illuminating element on the viewer that illuminated an extended element was proposed [81]. A set of reference points on the base of the extended element allowed for pose determination by the viewer by looking at the extension of the shadow, similar to a sundial.

Bruckstein et al. proposed a number of unpowered encoding elements in [67], based on serrated surfaces that appear to change "greyness" based on the viewing angle, a sundial fiducial similar to the one proposed by Kunkel in [81] and a fiducial based on the eye of a praying Mantis.

Tanaka et al. have provided a range of different versions of encoding elements utilising lenticular lenses since 2012 (shown in Figure 5.2 [60, 82, 83]. The fiducials that utilised these elements are called Lentimark by the authors. These encoding elements utilise an array of lenses on a stripe pattern, producing the effect of a travelling stripe with respect to the relative pose of the viewer. The encoding elements solve the problem of pose ambiguity as well as the problem of planar fiducials with respect to the loss of precision in frontal observation. However, these elements only enable retroactive pose correction, increasing the pose estimation algorithmic complexity.

In addition, in 2012 Tanaka et al. proposed two-dimensional version of the Lentimark encoding element, based on an array of microlenses with a cross pattern in 2 directions called "ArrayMark" [84, 85]. This encoding element has a moving cross in two dimensions, similar to using two lenticular elements. Also similar to the Lentimark, the encoding elements only enable retroactive pose correction.

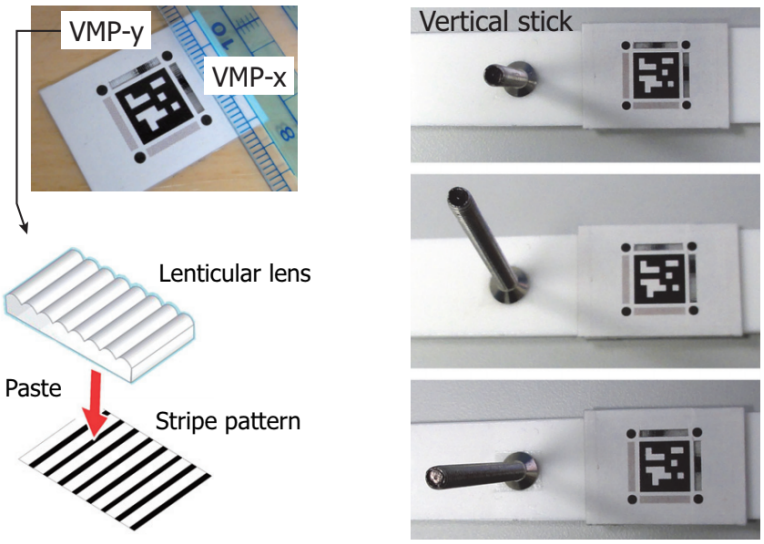


Figure 5.2: Lentimark principle. Reprinted from [60], with permission. ©2014 IEEE.

Similar to Tanaka et al., Schillebeeckx et al. proposed using a coloured pattern in conjunction with lenticular lenses to create a fiducial element that changes colour based on the relative pose of the viewer [86, 87]. A similar proposal was patented in [88]. The advantage of the Schillebeeckx et al. marker with respect to the Lentimark is that due to the encoding of the relative pose in colour instead of a moving element the required surface area for the encoding element is smaller. However, the authors report a slightly lower accuracy and precision performance with respect to the Lentimark. Also, the use of colour as encoding medium can be challenging in changing illumination conditions. In [89], Xuan, Schillebeeckx and Pless proposed an algorithm that allows self-calibration of the colour signal deviations between these fiducials due to manufacturing challenges.

In 2016, Schillebeeckx and Pless proposed an encoding element that hashes the relative pose of the viewer using an array of microlenses and a bit pattern underneath the lenses [90]. This fiducial is shown in Figure 5.3. It uses a combinatorial encoding for the pose. The authors report similar performance to Lentimark, ArrayMark and the colour encoding fiducials by Schillebeeckx et al. The design is claimed to be advantageous to existing encoding elements since its design is flexible in application and can even consist of distributed lenses over a plane.

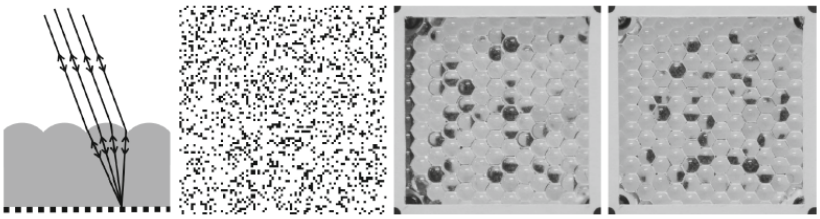


Figure 5.3: Schillebeeckx and Pless microlens fiducial. To the right is the viewed bit pattern under two distinct relative poses. Reprinted from [90], with permission. ©Springer International Publishing AG 2016.

5.2. Iterative Design Approach

The design of the Mantis Marker has been based on an iterative design approach. To converge to the final solution as it is presented in section 5.3, a number of iterations have been performed. For the sake of brevity, not all iterations of the design are shown here. However, the approach and tools used to perform the iterations do require some justification.

Each iteration of the design started with a system concept of the encoding element. These concepts were then transferred to a CAD drawing in Solidworks. This allowed a detailed design with relative dimensions. This design was then tested virtually using the Blender model described in section 7.2 and the feature extraction software and pose estimation software written at that specific iteration. After testing the design, improving software and checking for unexpected failure modes, another strawman concept was drawn and the design process re-iterated. Schematically this process is shown in Figure 5.4.

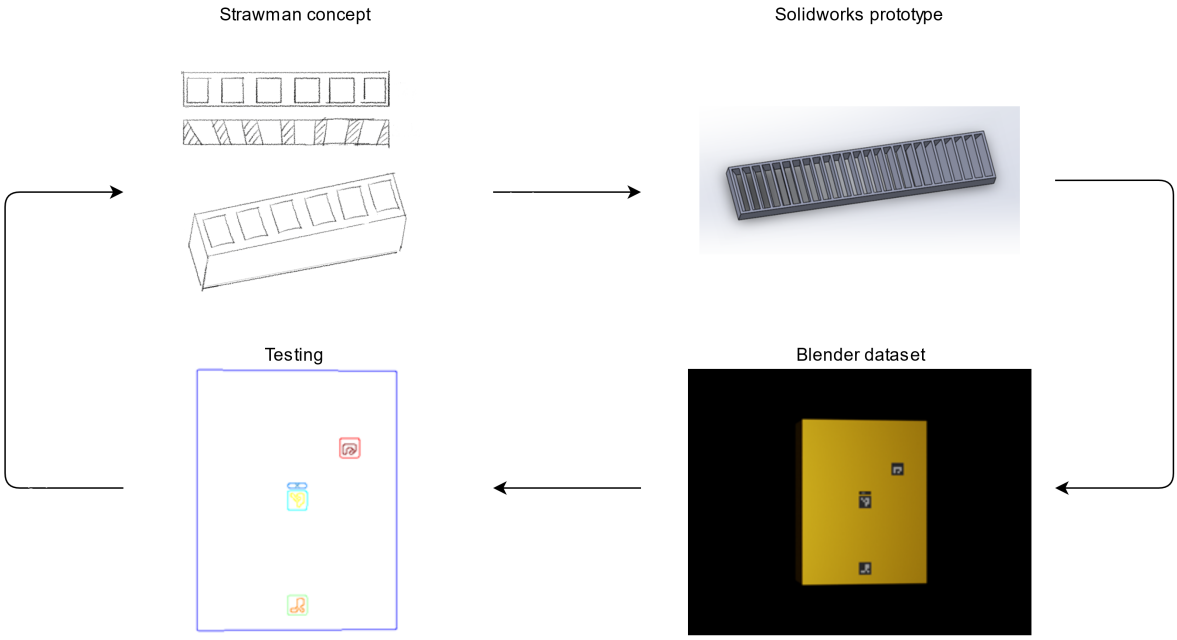


Figure 5.4: Iterative approach for designing the Mantis Marker.

Additionally, some marker designs were manufactured using 3D printing to validate at a high level the behaviour of the encoding element and to communicate design progress. Some of these markers are shown in Figure 5.5.

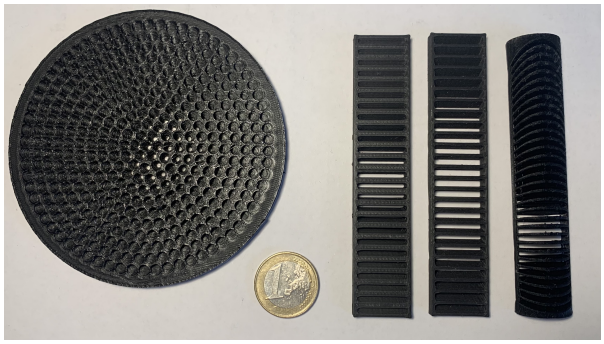


Figure 5.5: 3D printed iterations of the Mantis Marker, in no particular order. To the left is a (rejected) two dimensional implementation of the marker.

For the feature extraction software and as a proof of concept of the working principle of the Mantis Marker, a "Minimum Viable Product (MVP) approach" was used. After generating a dataset of the initial encoding marker design using Blender, images of this dataset were transferred to Matlab. In Matlab, an initial feature extraction algorithm was written to test the feasibility of the Mantis Marker working principle and to test which feature extraction algorithms were applicable. Then, the correlation between the rotation of the element in the pitch direction and the recorded signal was measured, shown in Figure 5.6. This initial iteration validated the proof of concept of the Mantis Marker principle and provided a MVP for both the fiducial and the software.

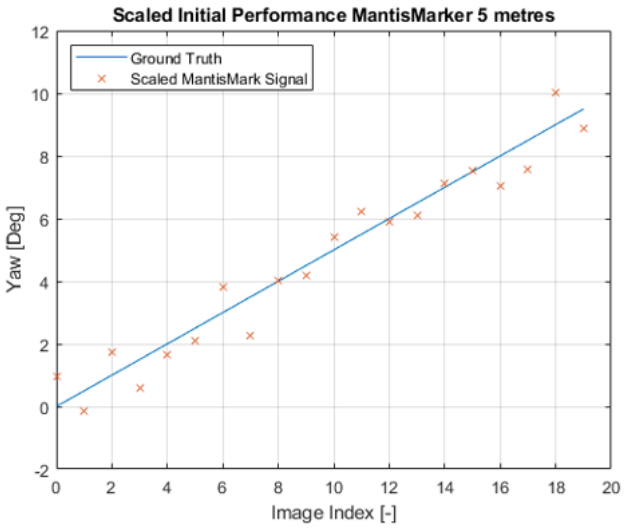


Figure 5.6: Result of the correlation analysis between pitch and encoding fiducial signal of the MVP

After this initial software implementation, all software was entirely redesigned and rewritten in C++. Pose estimation software was added to the implementation. All required software iterations were from the MVP onward programmed exclusively in C++.

5.3. Design

As stated in chapter 4, the tradeoff analysis for the concept selection for the encoding element showed that no current encoding element is able to meet the posed requirements for the RENDER system. Therefore, a new type of encoding element needs to be designed.

As stated in section 5.1, Bruckstein et al. propose a fiducial based on the eye of a Mantis in [67]. A Mantis eye has an apparently moving black spot, following the viewer, called a pseudopupil. This pseudopupil is shown in Figure 5.7.

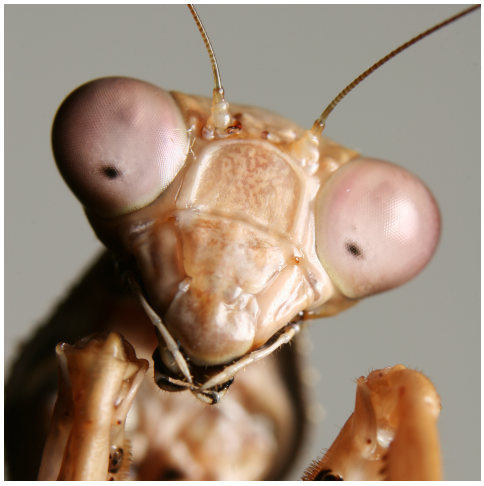


Figure 5.7: Eye of a Mantis, showing the pseudopupil. Image by Steve Smith, public domain.

While apparently moving, there is no physical movement of a pupil in the insects' eye. The apparent movement is caused by the geometry of the eye, which consists of long cylinders called *ommatidia*. This principle is shown in Figure 5.8. Only in the ommatidia that are viewed in line with the line of sight of the viewer, the

bottom of the ommatidia can be seen. Since this bottom absorbs light, no light is reflected in this direction causing the black spot [91].

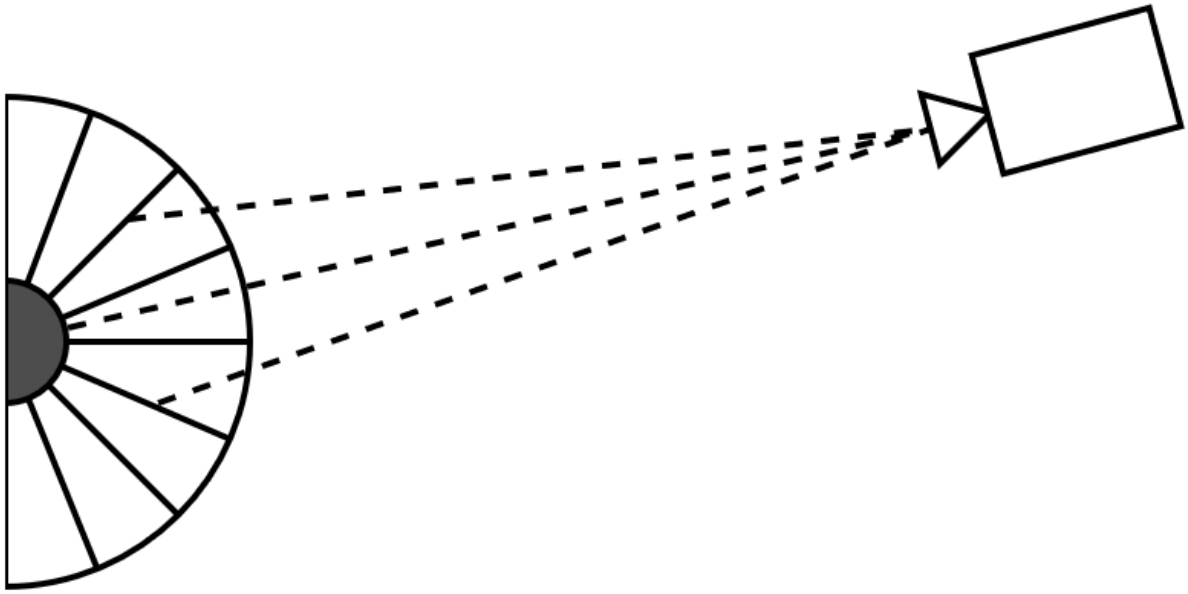


Figure 5.8: Principle of the pseudopupil. The black "pupil" is only visible through the ommatidia that are aligned with the viewer. Based on [67].

The idea of using a similar principle is interesting for use in pose estimation fiducials: it encodes rotations in a small form factor. However, to the knowledge of the author of this thesis, this idea was never developed beyond the proposed concept. Furthermore, the concept proposed by Bruckstein et al. is complex and non-passive. The authors propose the use of a light source at the centre of the fiducial and optical elements that allow only certain wavelengths of light to pass, making the fiducial change colour when perceived from different relative angles. Apart from complexity, this proposal does not allow for scalability. In addition, while the authors do not propose any concrete designs, the planarity requirement of RENDER requires a flat marker, discarding any semi-circular designs.

Thus, a different approach is chosen for RENDER. Instead of using colour as the encoding medium, it is for many applications more beneficial to use a spatial signal to encode pose information (especially applications with adverse or unpredictable illumination conditions such as the space environment). Realising this, it is also necessary to flatten and elongate the "eye". Since the information will be encoded in a spatial signal, there needs to be enough resolution on the image plane to do so which causes the need for an elongated marker. Furthermore, the design of the Mantis eye as it appears in nature is *inverted*: the encoding element has a reflective element at the bottom of an ommatidium and absorbing elements around the edges. This produces a reflective pseudopupil that appears on a black background. As an added benefit, it removes the need for any optical or active components on the encoding element. It does however impose an additional requirement on the viewer: the viewer should have an illumination source parallel with the viewing direction to ensure proper illumination of the reflective ommatidium bottom.

The final design of the encoding element is shown in Figure 5.9. Based on the eye of a Mantis, it is a flat and one-dimensional interpretation of the insect eye. The encoding element does not use any optical element such as lenses, is scalable and can be made out of 1 part. It is 3D printable and due to the simple geometry simple to manufacture by other techniques such as milling. When viewed with appropriate illumination from the viewing direction, a "pseudopupil" or blob encodes pose information by its centroid position on the encoding element.

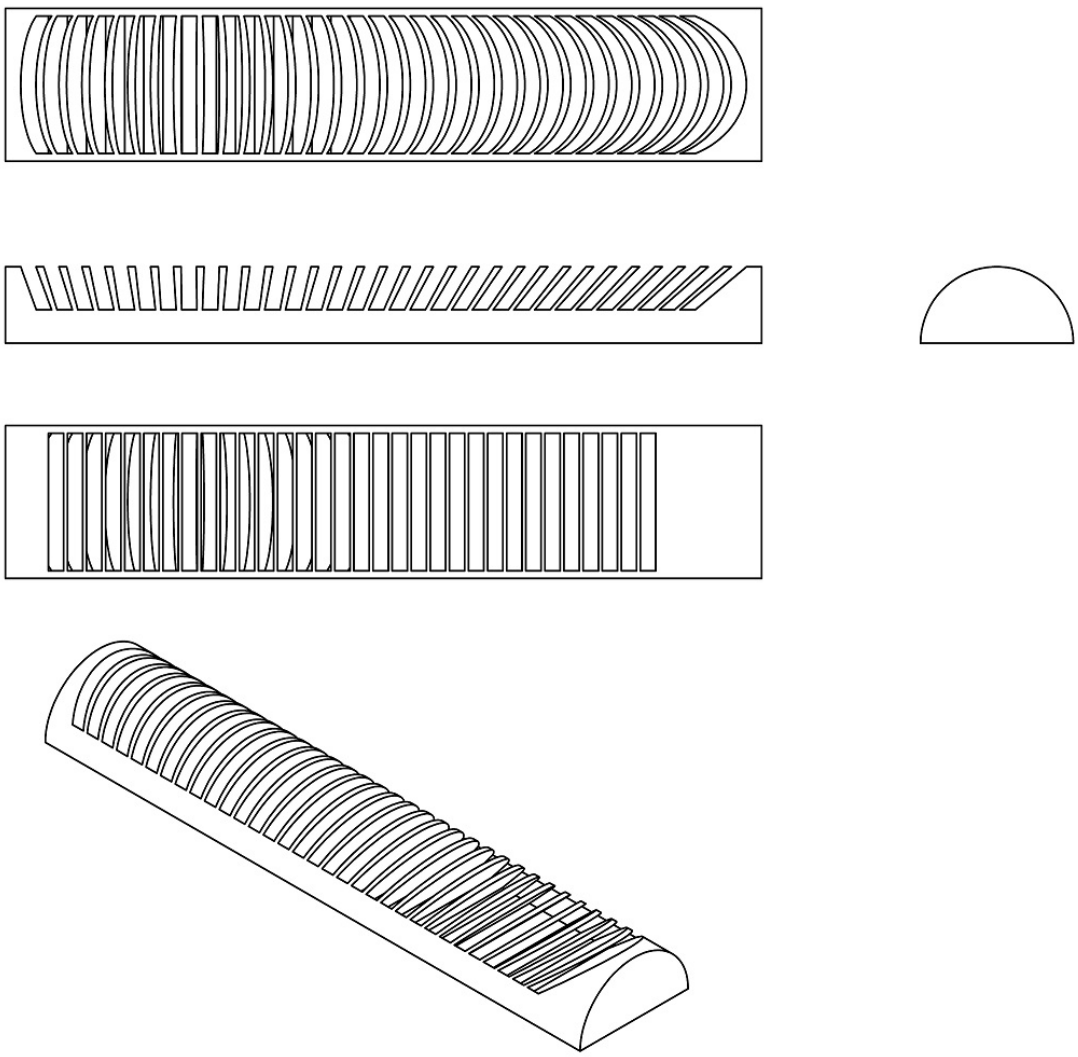


Figure 5.9: Design drawing of the encoding element of the Mantis Marker. Note that this drawing does not include any dimensions: the marker is inherently scalable and thus not constrained by dimensions.

The Mantis Marker encoding element has another unique feature that, to the knowledge of the author, no existing encoding element or indeed any fiducial marker has. Due to its geometry, the marker does not only encode a function of rotation in its signal, it actually encodes a virtual point in space. This is shown in Figure 5.10. This feature has several implications for the relative pose estimation using the Mantis Marker. Since the element encodes an actual point in space, the projection of this point on the element's "plane of projection" and subsequent projection on the image plane is equivalent to the projection of an actual point at the physical location of the virtual point.

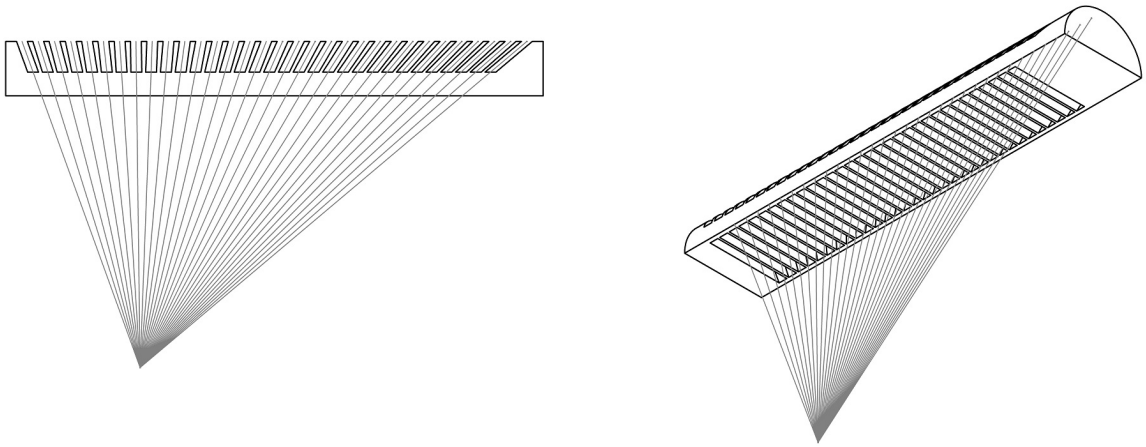


Figure 5.10: Design drawing showing the virtual point of the Mantis Marker. The virtual point is placed off centre, to increase the FOV of the fiducial marker implementation (as explained in section 5.6).

This is shown in Figure 5.11. In other words, the encoding element of the Mantis Marker is *equivalent* to using a marker with a protrusion, such as the SVMS. The virtual point can directly be used as a point in space in the PnP algorithm. This is opposed to current encoding fiducials such as Lentimark, which retroactively correct the calculated pose by using an approximation function of the encoding signal with respect to the relative pose [82].

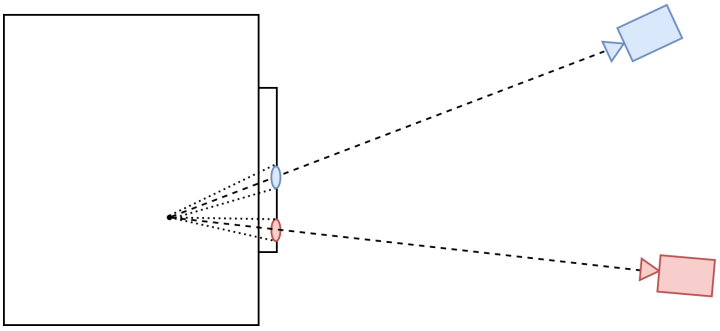


Figure 5.11: Working principle of the virtual point. The Mantis Marker projects a physical location in space at depth on the image plane without requiring an actual protrusion to that location.

In order to provide a 6-DOF pose estimate, an existing planar fiducial is added to the encoding elements. For this fiducial, the Aruco marker [5] is chosen since its implementation readily available in the open-source OpenCV software library[1] and it is widely used. Note that the Mantis Marker implementation is not limited to this fiducial, other equivalent fiducials could be used as well. The complete fiducial marker is shown in Figure 5.12. White or reflective borders are added to the encoding elements as well as the Aruco marker to aid in the corner detection required by the feature extraction pipeline. Four encoding elements are added. Two encoding elements make up a virtual point in three-dimensional space (one per dimension on the two-dimensional image plane). With the addition of 2 more encoding elements with off-centre virtual points, the

[1] Available at: https://opencv.org/

FOV of the marker is greatly increased. With this design, a total of four virtual points (equal to the number of possible combinations of the two horizontal and vertical elements) are added to the four real points of the Aruco marker. The materials used for the Mantis Marker will be discussed in chapter 6.

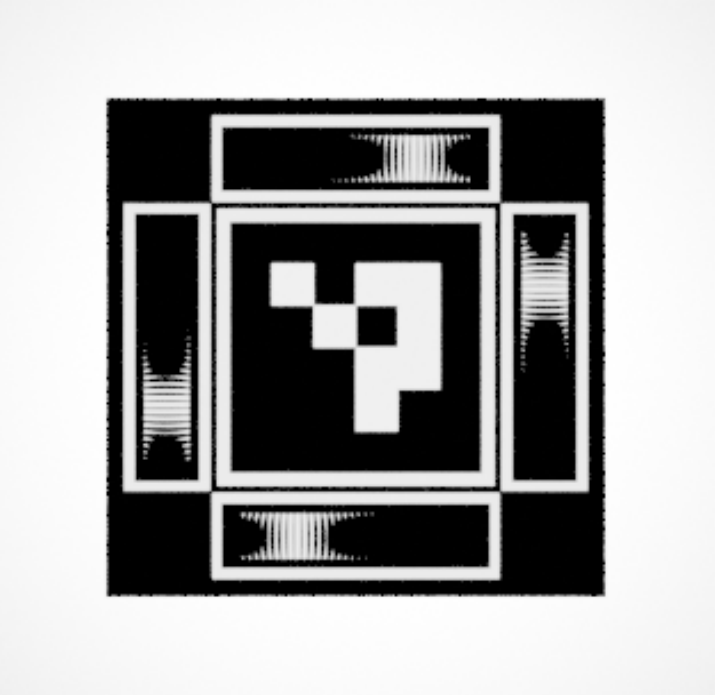


Figure 5.12: Render of Mantis Marker with Aruco implementation.

5.4. Computer Vision Pipeline

All written code for the computer vision pipeline and the pose estimation software can be found in Appendix A.

A fiducial cannot be designed on its own: it is part of an end-to-end navigation system. The required software for this system can be regarded as a subsystem that takes as input an image of a fiducial, extracts the required features from this image and passes these features to a pose solver which outputs the pose. This section details the feature extraction software for the Mantis Marker system and section 5.5 details the pose estimation software. All software has been written in C++ using the OpenCV library.

At the highest level, the feature extraction software for the Mantis Marker can be described by a block that has as input an image of the scene in which there may be any number of markers and as output all the positions of the corner points of the Aruco and virtual points of the encoding elements in the image plane.

In Figure 5.13, a functional flow diagram of the feature extraction software is shown. The first step in the feature extraction algorithm is the detection of the Aruco marker(s) in the scene using the standard Aruco function cv::aruco::detectMarkers[2]. If no Aruco is found, the feature extraction fails and an error is thrown. If an Aruco is found, the known geometry of the Mantis Marker fiducial is used to calculate a contour for all 4 encoding elements. For this, a custom helper function is used called find_markers. This function takes as input the location of the four Aruco corners on the image plane and a "width factor" and "gap factor" which are the width of the encoding elements and the gap between the Aruco and the encoding elements divided by the Aruco size. The output of this function are the four corners of all four encoding elements.

[2]For reference: https://docs.opencv.org/master/d9/d6a/group__aruco.html. Acessed 1-5-2021

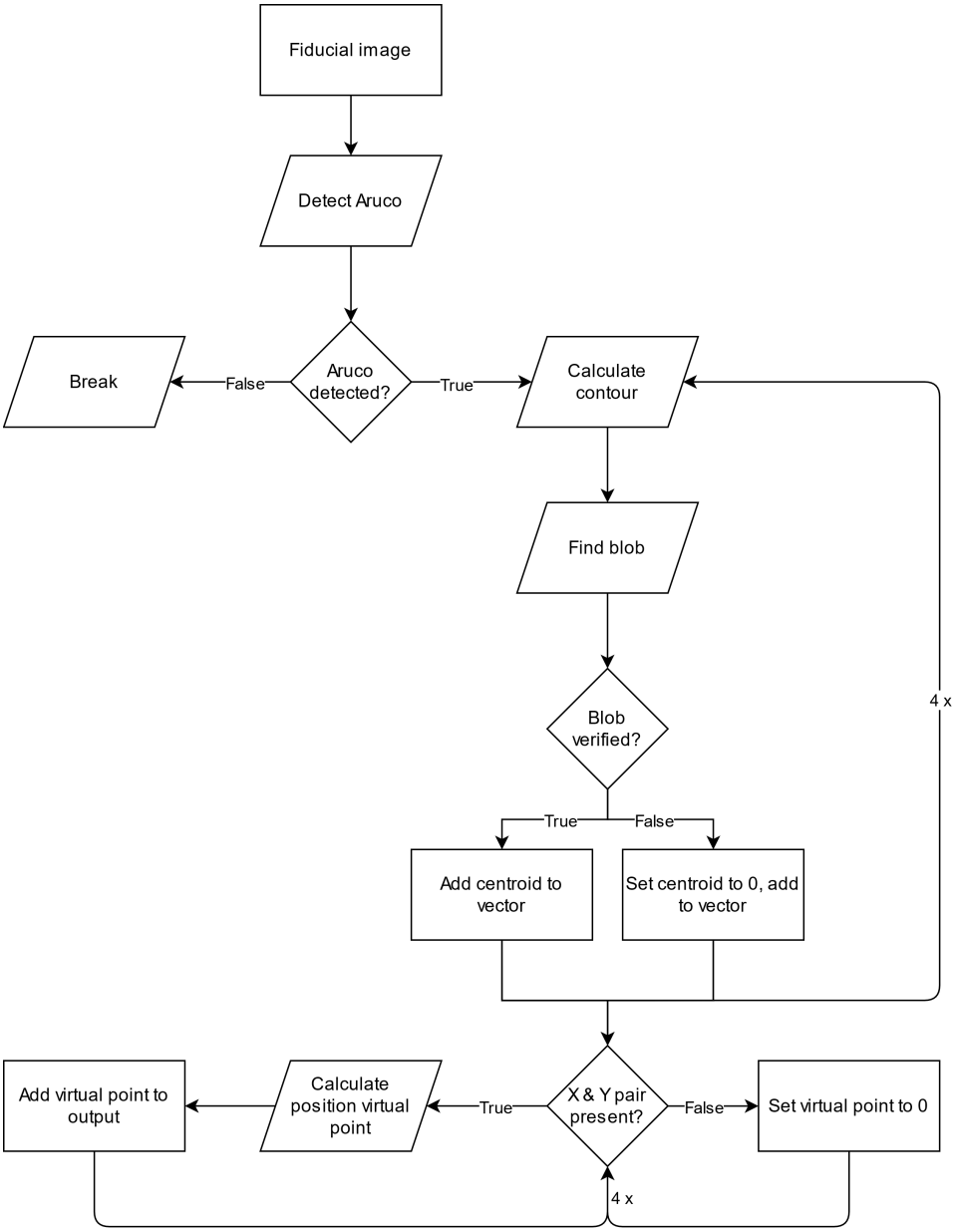


Figure 5.13: Top-level functional flow feature extraction.

These corners are calculated as follows:

$$m_1^i = c_j + (c_j - c_{j+3})(h + g - margin) \tag{5.1}$$

$$m_2^i = c_{j+1} + (c_{j+1} - c_{j+2})(h + g - margin) \tag{5.2}$$

$$m_3^i = c_{j+1} + (c_{j+1} - c_{j+2})(g + margin) \tag{5.3}$$

$$m_4^i = c_j + (c_j - c_{j+3})(g + margin) \tag{5.4}$$

$$\text{for } i = 1,2,3,4 \text{ and } c_j = \begin{cases} c_j \text{ for } j \leq 4 \\ c_{j-4} \text{ for } j > 4 \end{cases} \tag{5.5}$$

Where m_k^i is the scalar value of the x and y coordinates (respectively) of encoding element corner k (clockwise starting top left) for encoding element i (clockwise starting top left). c_j is the scalar value of the x and y

coordinates of the Aruco corners (respectively). h and g are the height of the encoding element and the gap between the encoding element and the Aruco divided by the Aruco width, respectively. *margin* is a margin added to leave a small border around the contour of the marker, currently set at 5%. The geometry of this problem is shown in Figure 5.14.

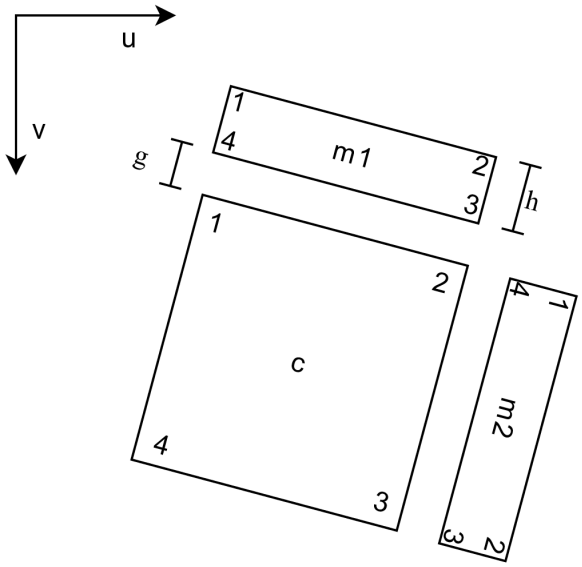


Figure 5.14: Geometry of the corner calculation algorithm.

Once the contours are defined a mask is generated for each encoding element, starting with the top element and moving clockwise (shown in Figure 5.15). For each encoding element, a Gaussian blur is applied to the mask to smooth out noise. Then, a blob detection algorithm (cv::SimpleBlobDetector[3]) is applied. This algorithm thresholds the image in a number of steps, calculating the centre of gravity for any white blob that may be present. After a predefined number of thresholds, all centroids are averaged to determine the overall blob centroid with subpixel accuracy. The blob is verified to comply with the expected area and inertia. Next, the blob centroid is verified to be located far enough from the edge of the encoding element, as being too close to the edge may cut off part of the blob size thus shifting the blob projection location from its actual location [92]. If the blob fails verification, a zero is passed to the next block in the algorithm. If the blob is verified successfully, the x and y coordinates are passed.

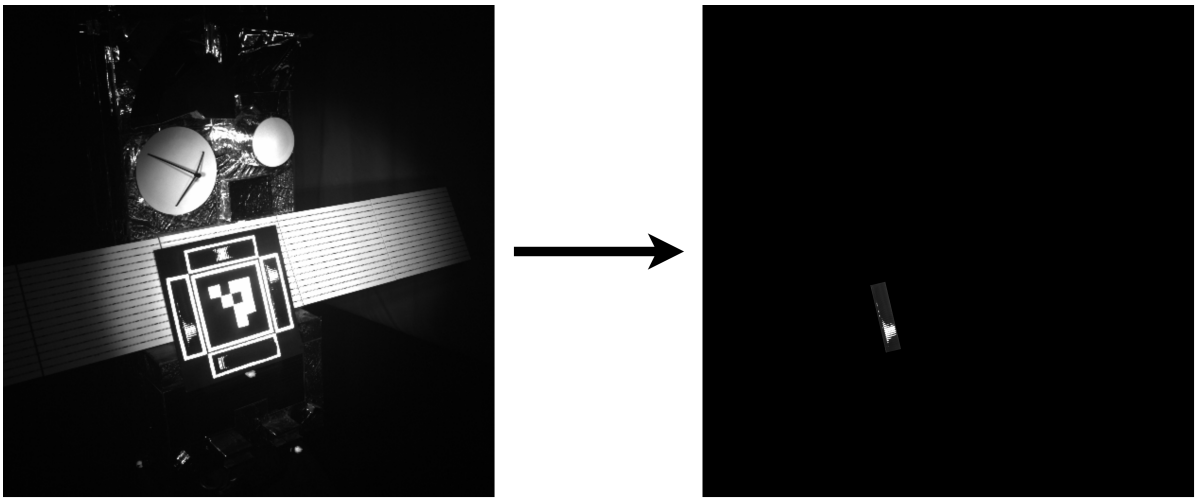


Figure 5.15: Mask generation from input image

[3]For reference: https://docs.opencv.org/3.4/d0/d7a/classcv_1_1SimpleBlobDetector.html, accessed 7-5-2021

When all 4 blob centroids are determined, the next step is to calculate the location of the virtual points on the image plane. A similar approach is used to the calculation of the mask for the encoding element. The slope of the Aruco sides are calculated, after which the following equations are applied for the virtual points for which the respective blobs were found:

$$v_i = a_v * u_i + b_j \tag{5.6}$$

$$v_j = a_h * u_j + b_j \tag{5.7}$$

$$b_i = v_i - (a_v * u_i) \tag{5.8}$$

$$b_j = v_j - (a_h * u_j) \tag{5.9}$$

$$u_k^v = \frac{b_i - b_j}{a_h - a_v} \tag{5.10}$$

$$v_k^v = a_v * u_k^v + b_i \tag{5.11}$$

$$\text{where } i = 1,3 \text{ and } j = 2,4 \text{ and } k = 1,2,3,4 \tag{5.12}$$

Where u_n and v_n are the horizontal and vertical coordinates of the nth blob in the image frame, a_v and a_h are calculated vertical and horizontal slopes of the Aruco, b_n is the y axis intercept of the line parallel to the relevant Aruco slope and crossing the blob centroid and u_k^v and v_k^v are the horizontal and vertical coordinates of the kth virtual point. The output of this calculation is the virtual point coordinates in the image frame. The output of the feature extraction function is plotted in Figure 5.16.

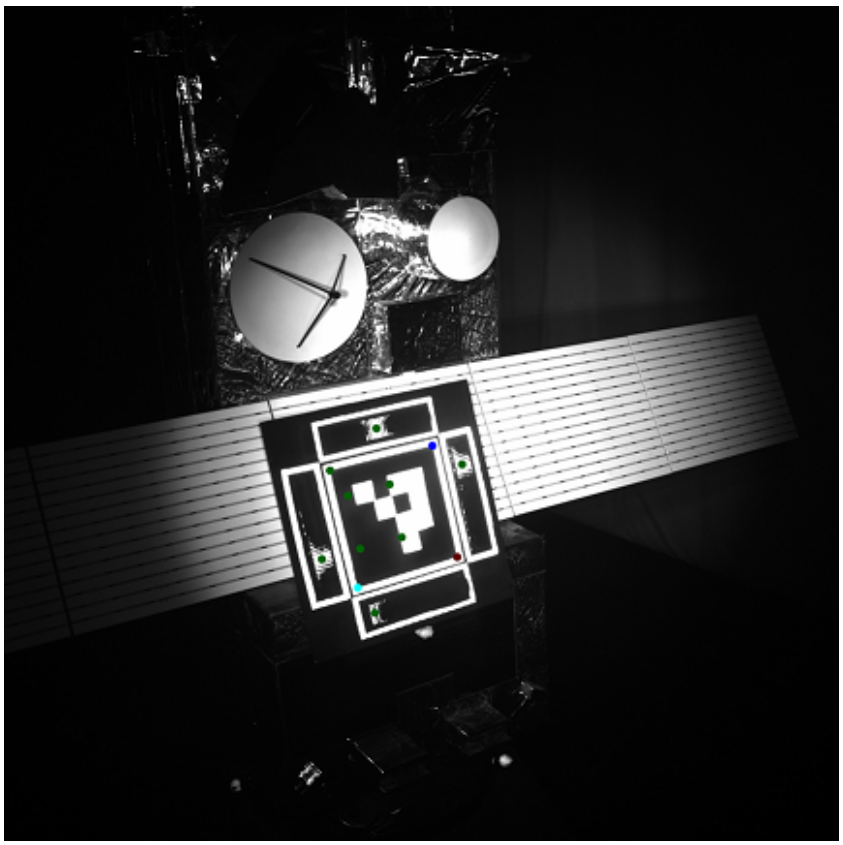


Figure 5.16: Output of feature extraction. Note that the bottom blob note complete due to being too close to the edge of the marker, thus both virtual points on the left are not be passed to the pose estimation block. Furthermore, the found blob centroids are not passed to the pose estimation block, only the Aruco corners and the virtual points.

5.5. Pose Estimation Software

The next function in the functional flow of the software is the determination of the pose corresponding to the found location of all points in the image frame (both the Aruco corner points and the virtual points).

This process is shown schematically in Figure 5.17. The software takes as input the found points, the dimensions of the Aruco as well as the location of the virtual points in the world frame (which are a function of the encoding element dimensions). Finally, the camera intrinsics of the calibrated camera are passed as input.

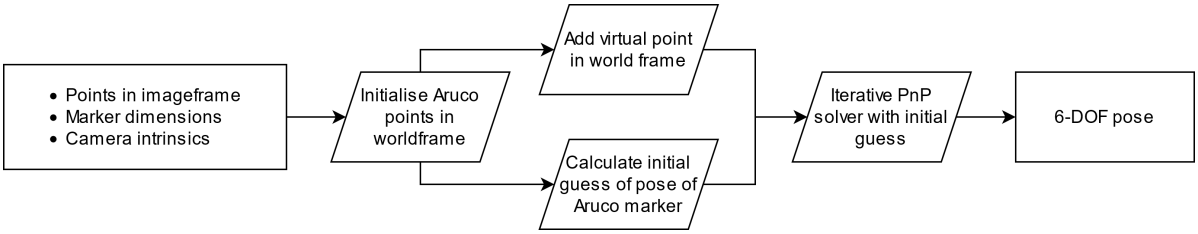


Figure 5.17: Top-level functional flow pose estimation software.

First, an initial solution to the pose estimation is found using only the 4 co-planar Aruco corner points. For this, the OpenCV function cvFindExtrinsicCameraParams2 is called, which in turns calls the function findHomography[4]. This function finds the homography matrix H. In homogeneous coordinates, H is defined such that:

$$s \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \tag{5.13}$$

Homography matrix H is the camera intrinsic matrix multiplied by the transformation matrix between the object frame and the image frame (see Equation 2.5). If the object is co-planar (as it is for the Aruco marker), z_i can be set to 0. This simplifies the equation as follows [93]:

$$H = I \begin{bmatrix} r_1 r_2 r_3 t \end{bmatrix} \tag{5.14}$$

$$s \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \sim I \begin{bmatrix} r_1 r_2 r_3 t \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 0 \\ 1 \end{bmatrix} \sim I \begin{bmatrix} r_1 r_2 t \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{5.15}$$

With this simplification, the H is a 3x3 matrix. Then an initial estimate for H can be found by minimising the reprojection error using a least-squares minimisation. The reprojection error is defined as:

$$error = \sum_i \left(u_i - \frac{h_{11} x_i + h_{12} y_i + h_{13}}{h_{31} x_i + h_{32} y_i + h_{33}} \right)^2 + \left(v_i - \frac{h_{21} x_i + h_{22} y_i + h_{23}}{h_{31} x_i + h_{32} y_i + h_{33}} \right)^2 \tag{5.16}$$

Where h_{ij} are the respective values in H. The initial estimate found is then refined using the Levenberg-Marquardt method. The output of this routine is the initial guess of the pose of the Aruco marker.

[4]For reference: https://docs.opencv.org/3.1.0/d9/d0c/group__calib3d.html, accessed on 8-5-2021

Next, the location of one of the found virtual points in the world frame is added to the object point list and together with the initial guess of the relative pose and the location of all points on the image plane passed to the PnP solver solvePnP[5]. With the given initial guess, this solver finds the relative transformation again by minimising the reprojection error (this time off all points including the virtual point) using a Levenberg-Marquardt method. In the current implementation, only one virtual point is added (even if more points have been found) to ensure constant reliability of results. This is of course adaptable per use case.

It should be noted, that the current implementation of the pose estimation algorithm for the Mantis Marker essentially involves two consecutive iterative PnP solvers, one with only the co-planar Aruco points to find an initial guess of the pose and one including the virtual point. This has been chosen for ease of implementation with the structure of the OpenCV library. In future work, this should be reduced to a single solver call. The current implementation shows that the Mantis Marker encoding element can be used directly in a PnP solving algorithm (without retroactive pose correction) and has the ability to prevent a PnP solver from finding (wrong) local minima in the reprojection error function. This is proven in chapter 8.

5.6. Encoding Element Parametric FOV Analysis

The FOV of the encoding element is dependant on several tuneable design parameters. To highlight some of the tradeoffs that are present in choosing the appropriate dimensions of the encoding element, some geometric parameters are developed into equations for the marker FOV here.

In general, with a larger distance between the virtual point and the plane of the encoding element (denoted by V_d in Figure 5.18), the "resolution" of the encoding element increases. In other words, the distance travelled by the blob on the image plane per degree is increased with an increased V_d. However, increasing this distance comes at a penalty of a lower FOV of the encoding element.

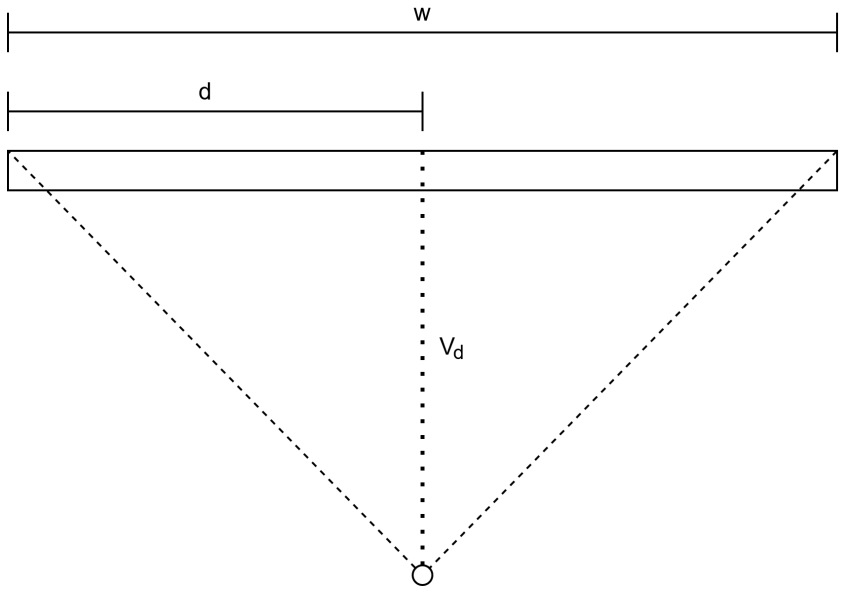


Figure 5.18: Schematic representation for FOV calculation of encoding element.

The FOV can be expressed as a function of the width w of the encoding element and the depth of the virtual point V_d. This can be expressed as follows:

$$FOV = tan^{-1}\left(\frac{w}{V_d}\right) \tag{5.17}$$

[5]For reference: https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html, accessed 08-05-2021

Note that in the equation above, a margin should be taken from the FOV in order to account for some distance from the edge of the marker where blob cutoff occurs. In the encoding elements used in the presented setup of the Mantis Marker, the virtual point has been moved from the centre of the marker. This has been done to make the FOV asymmetrical, which allows overlap between the FOV of the two parallel encoding elements. This increases the overall FOV of the entire encoding marker system. Now the equation for the FOV becomes:

$$FOV = tan^{-1}\left(\frac{d}{V_d}\right) + tan^{-1}\left(\frac{w-d}{V_d}\right) \tag{5.18}$$

For the current configuration, $d = 0.25w$ and $V_d = 2d$. This gives the entire encoding element a FOV of 82.9 degrees. Measured from the side to which the virtual point has been moved, the marker has a FOV of 26.6 degrees. On the long side, this is 56.3 degrees. An estimated 18 degrees is conservatively taken as a margin from both sides of the encoding element, making the FOV per element 46.9 degrees. With the current design of the Mantis Marker, 2 parallel encoding elements are oriented in the opposite way. This results in an overlap in FOV between the two elements of 17.2 degrees when taking the margin into account. Then, the total FOV of the Mantis Marker in its current configuration in the longitudinal direction of the encoding elements is:

$$FOV = 46.9^\circ * 2 - 17.2^\circ = 76.6^\circ \tag{5.19}$$

In the lateral direction of the encoding element, the round shape of the profile of the encoding element prevents the area of the blob to reduce when viewed at a more slanted angle. This is visible in Figure 5.16, the blob becomes wider when viewed from the side due to the anvil-shaped contour. It is estimated that the lateral FOV is around 100 degrees in the current configuration.

Combining these rotations gives an expression of the FOV of an encoding element as a function of pitch and yaw:

$$1 = \frac{\psi^2}{FOV_{lat}^2} + \frac{\theta^2}{FOV_{long}^2} \tag{5.20}$$

Where ψ is the yaw defined in the lateral direction of the encoding element and θ is the pitch defined in the longitudinal direction of the encoding element. In this case. FOV_{long} is the longitudinal FOV of the particular side of the encoding element where the FOV is calculated.

5.7. Conclusion

Pose estimation using planar fiducials causes performance issues such as loss of precision in frontal observation and pose ambiguity. To mitigate these issues, an encoding fiducial may be used. However, existing encoding fiducials are highly complex and not feasible for use in space. Therefore, a new type of encoding fiducial is proposed called the Mantis Marker.

The Mantis Marker consists of four encoding elements as well as a standard planar fiducial. The planar fiducial ensures unique identification of the marker and adds four co-planar points to the pose estimation input. The encoding elements add up to four virtual points at depth, enabling pose estimation using the planar Mantis Marker to be mathematically equivalent to having a fiducial with protrusion. The Mantis Marker was designed using a highly iterative approach.

The encoding element design is based on the eye of a Mantis. These insects have eyes that contain *ommatidia* that cause the projection of an apparently moving pseudopupil. This principle is used to project the virtual point at depth using the encoding elements. The encoding elements in the end-to-end fiducial have an overlapping FOV due to off-centre virtual points, increasing the overall system FOV.

To extract features from the Mantis Marker, a computer vision pipeline was written. This pipeline is based on the OpenCV library and uses a series of algorithms to find the blob locations of the encoding elements with subpixel accuracy. From these blob locations, the location of the four virtual points on the image plane is calculated. The pose estimation software takes the features found by the feature extraction pipeline to initialise an iterative pose solver which outputs the relative pose between camera and fiducial.

6

End-to-End Relative Navigation System for Space Applications Using the Mantis Marker

The Mantis Marker is a powerful fiducial system for efficient relative pose estimation. However, implementing this system in an end-to-end relative navigation for debris removal requires additional analysis. A number of questions need to be answered, e.g.: how many markers need to be present on the baseline spacecraft defined in chapter 3 in order to fulfil the requirements? What dimensions should the markers have? What would be an appropriate material? How should the fiducial marker system be illuminated?

To answer these questions, two system layout design options are proposed and some additional placement considerations are listed in section 6.1. Next, the system illumination is sized and a radiometric analysis of the end-to-end system is performed in section 6.2. Finally, some potential materials are mentioned and the effects of the space environment are briefly covered in section 6.3.

6.1. System Layout

The main driver for the determination of the required number of Mantis Markers and their respective spacing, sizes, camera FOV and detector resolution etc. is the required accuracy and precision requirements posed in chapter 3. Furthermore, some constraints in terms of spacing, marker size and total system footprint were posed which constrain the possible system layouts.

In order to find the appropriate system layout, some assumptions are made to provide a baseline for the system model. First of all, in accordance with [SYS-REQ-11], the minimum spacecraft face dimensions (1 x 1 metre) are chosen since this is the most conservative (i.e. smallest) required face dimension. The maximum individual marker dimensions then follow from [SYS-REQ-6], being 0.1 x 0.1 metres. The total combined planar area of the individual markers may not exceed 0.025 m^2 in accordance with [SYS-REQ-7].

To analyse the sensitivity of the fiducial marker system to the placing of fiducials on a spacecraft face, two typical placement profiles are analysed (shown in Figure 6.1). The first option is a widely distributed marker placement profile, called the "wide" option. In the centre of the spacecraft face, four smaller markers are placed close together. Since the chaser spacecraft is assumed to approach the spacecraft to the centre of this face, this ensures enough points are in the FOV of the camera at short range. The other markers are placed near the corners of the spacecraft (leaving some margin for e.g. attitude control thrusters). The inter-marker distance is thus maximised, which, in theory, should increase performance at range.

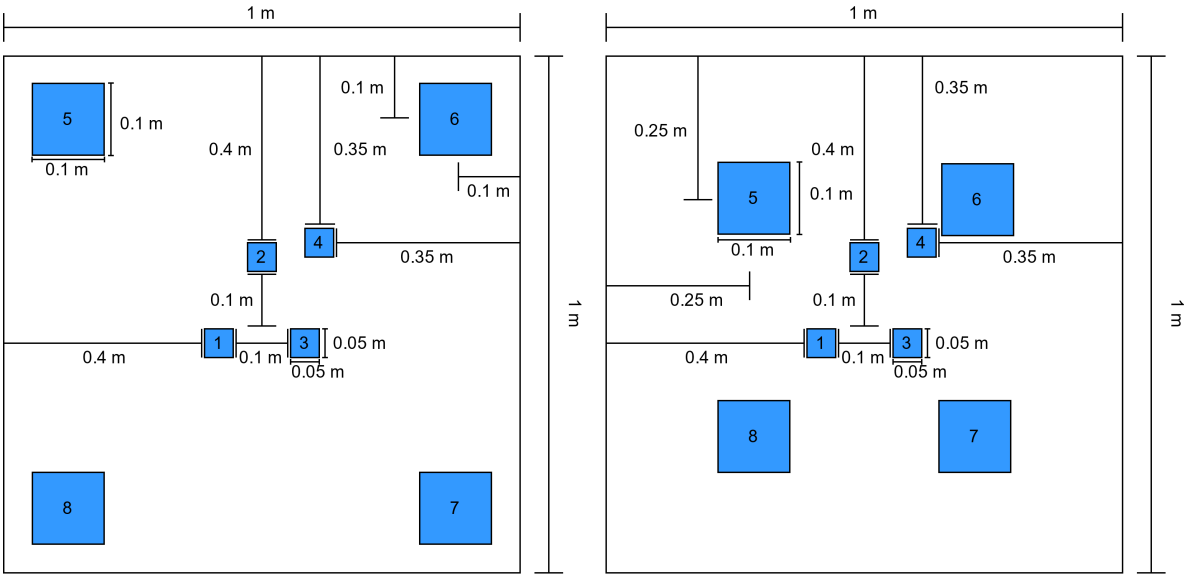


Figure 6.1: Typical placement options for the fiducial markers on a the smallest applicable spacecraft face. On the left is the "wide" option, on the right the "narrow" option.

The second placement option to be analysed is a more narrow option, shown on the right in Figure 6.1. The inter-marker distance is reduced in this placement option, thus reducing performance at range. The performance of these two design options is modelled using a Monte Carlo simulation in chapter 8.

An additional consideration in the layout of the end-to-end fiducial marker system is the necessity to prevent an approach trajectory that causes the Sun to be behind the target object face that is used for relative navigation in the approach. A logical design choice would be to always have two opposing marked spacecraft faces. If the spacecraft is model as an entirely symmetrical cube, two perpendicular spacecraft axes with two opposing marked faces (i.e. 4 spacecraft faces in total) is necessary. This prevents all marked spacecraft faces to be around the axis of rotation. If the spin axis of the target spacecraft in an uncontrolled state is somehow predictable due to the shape of the object, this could reduce the number of required marked faces to two on opposing sides.

6.2. System Illumination and Radiometric Analysis

The challenging illumination environment in space causes a need for illumination of the fiducial marker system by the target spacecraft. To ensure robustness of the system level performance under all relevant conditions, both very dark (eclipse) and very bright (sunlight) use cases are considered. In eclipse, the system should be able to provide sufficient illumination to extract the features for the PnP algorithm (at an appropriate signal to noise ratio). In sunlight, the system should be able to extract the same features while preventing over-saturation of the detector as well as filtering spurious reflections. For both illumination cases, any secondary illumination sources (such as Earth albedo) are disregarded in this analysis.

To deal with both extremes, it is decided to attempt to reduce the delta between two illumination environments as much as possible. To this end, a bandpass filter is applied to the camera system. The idea behind this approach is that the bandpass filter is able to filter out a significant portion of the sunlight, thus reducing the impact of sunlight on the system. In addition, an infrared illumination source is chosen to *maximise* the delta between sunlight and the illumination system. In qualitative terms, this is shown in Figure 6.2. This system should then be sized in terms of illumination power.

The limiting case for the illumination is naturally at the largest distance between chaser and target, which is set by the system requirements at 20 metres. It is assumed that the bandpass filter that is applied to the

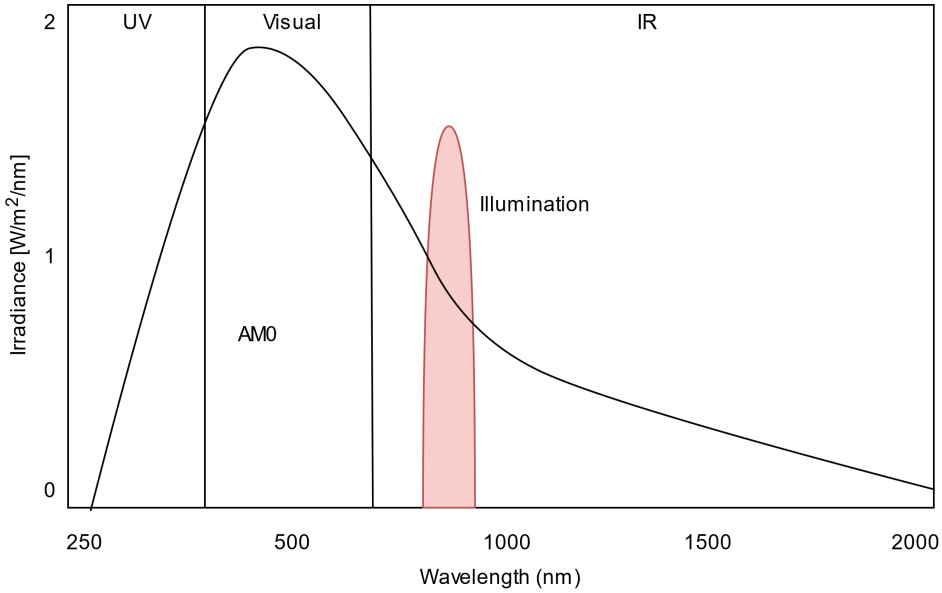


Figure 6.2: Qualitative plot of illumination wavelength versus sunlight. Naturally, it is not possible nor required for the system to achieve a higher irradiance than the Sun at all ranges as depicted here. However, in order to mitigate the influence of a significant portion of the sunlight, the wavelength of the illumination should have the largest feasible delta.

camera is a highpass filter that allows light with a wavelength of 800 nm or higher to pass. In addition, the quantum efficiency of the detector is assumed to be 35% for 800 nm to 900 nm, and 0% for all light above 900 nm. This is an approximation of the quantum efficiency of the Prosilica GT4096 NIR detector [94], used in the experimental setup in chapter 7. For the illumination, a light source with a wavelength of 850 nm is assumed to be used (again, identical to the used illumination in the experimental setup). For the camera system, an FOV of 16 x 16 degrees is taken with 752 x 752 pixels on the detector and a pixel size of $8.6 * 10^{-3}$ meters (similar to the benchmark VBS system used in chapter 3). An f-number of f/2.8 is assumed with a focal length of 20.19 mm and a throughput of 0.35.

The signal to noise ratio is calculated by taking the signal to be the light coming from a single *bit* in the bit pattern of the Mantis Marker (with approximately equal size to the encoding element blob). The noise is taken to be the total (shot noise limited) noise of the total signal (including reflections from the Multi Layer Insulation (MLI) around the marker). All calculations below are shown for the reflectivity of the bit but are applied in the same fashion to the reflectivity of the MLI around the bit. This parameter is taken, as it is the smallest optical element that needs to be distinguishable in order to identify the marker and extract features. These encoding elements have an approximate size of 1/10th of the marker dimensions. The dimensions are assumed to be 0.05 x 0.05 metre for the marker and 0.005 x 0.005 metre for the bit.

To simplify this analysis, the solar spectral radiance is modelled at 1 W/m^2/nm for the range of 800 nm to 900 nm. Since the camera only records light in these wavelengths, the solar irradiance E is calculated using Equation 6.2 to be 100 Watt/m^2:

$$E_{Sun} = \int_{800}^{900} 1 d\lambda \tag{6.1}$$

Where λ is the wavelength of light in nm and A_{marker} is the area of the marker in m^2. The light the marker receives is dependant on the illumination from the spacecraft as well as the illumination from the Sun (if present). For the illumination of the spacecraft, a typical infrared LED is selected called SFH 4783 - GX[1]. The irradiance for this LED is assumed to be spread over the full FOV and scaled accordingly. Then, 100 LED's

[1]Datasheet available at: https://www.digikey.nl/htmldatasheets/production/1762955/0/0/1/sfh-4783.html, accessed 21-05-2021

are taken and modelled as a point source, giving a radiant intensity $I_{radiant}$ of 40 kW/sr. In order to then calculate the irradiance of the bit by the illumination of the chaser, Equation 6.2 is applied.

$$E_{leds} = I_{radiant} * \Omega_{bit} \tag{6.2}$$

Where Ω_{bit} is defined as the solid angle of the bit as seen from the illumination:

$$\Omega_{bit} = \frac{w_{bit}^2}{R_i^2} \tag{6.3}$$

Where w_{bit} is the width of the marker bit in metres. With the irradiance calculated, the radiance from the bit is calculated to model the signal that is incident on the detector. For this radiance, the reflectance of the bit material needs to be assumed. Marchant et al. research the reflectivity of a number of retroreflective tapes in [95]. Based on their work, a conservative estimate of the Bidirectional Reflectance Distribution Function (BRDF) at normal incidence of this type of material is 200 sr^{-1}. Using these assumptions, the radiance L_{bit} can be calculated (shown in Equation 6.2). L_{bit} is given in Watt/m^2/sr.

$$L_{bit} = (E_{Sun} + E_{leds}) * BRDF \tag{6.4}$$

The irradiance per pixel by the bit is then calculated as follows:

$$E_{bit} = \frac{L_{bit} * A_{bit} * \Omega_{camera}}{n_{pixels,bit}} \tag{6.5}$$

Where Ω_{camera} is calculated using the diaphragm of the camera. For each range R_i the number of pixels per image of the bit n_{pixels} can be calculated as follows:

$$n_{pixels} = \left(tan^{-1}\left(\frac{w_{bit}}{2 * R_i}\right) * \frac{2}{fov}\right)^2 * A_d \tag{6.6}$$

Here, A_d is the area of the detector. The flux density Ψ in photons/second per pixel can then be calculated by:

$$\Psi = E_{bit} * A_{pixel} * \frac{\lambda}{h_{Planck} c} \tag{6.7}$$

Where λ is the wavelength of the illumination, h_{Planck} is the Planck constant and c is the speed of light. Then the signal S in electrons per pixel from the detector can be calculated using Equation 6.2:

$$S = \Psi * t * Qe * T_p \tag{6.8}$$

Where t is the exposure time, Qe is the quantum efficiency of the detector and T_p is the throughput. As stated, it is assumed that the system is shot noise limited. Therefore, the noise is calculated as:

$$N = \sqrt{S} + N_{shot,dark} \tag{6.9}$$

Here, $N_{shot,dark}$ is taken to be 28.2 electrons, in accordance with the Prosilica GT4096 NIR detector [94].

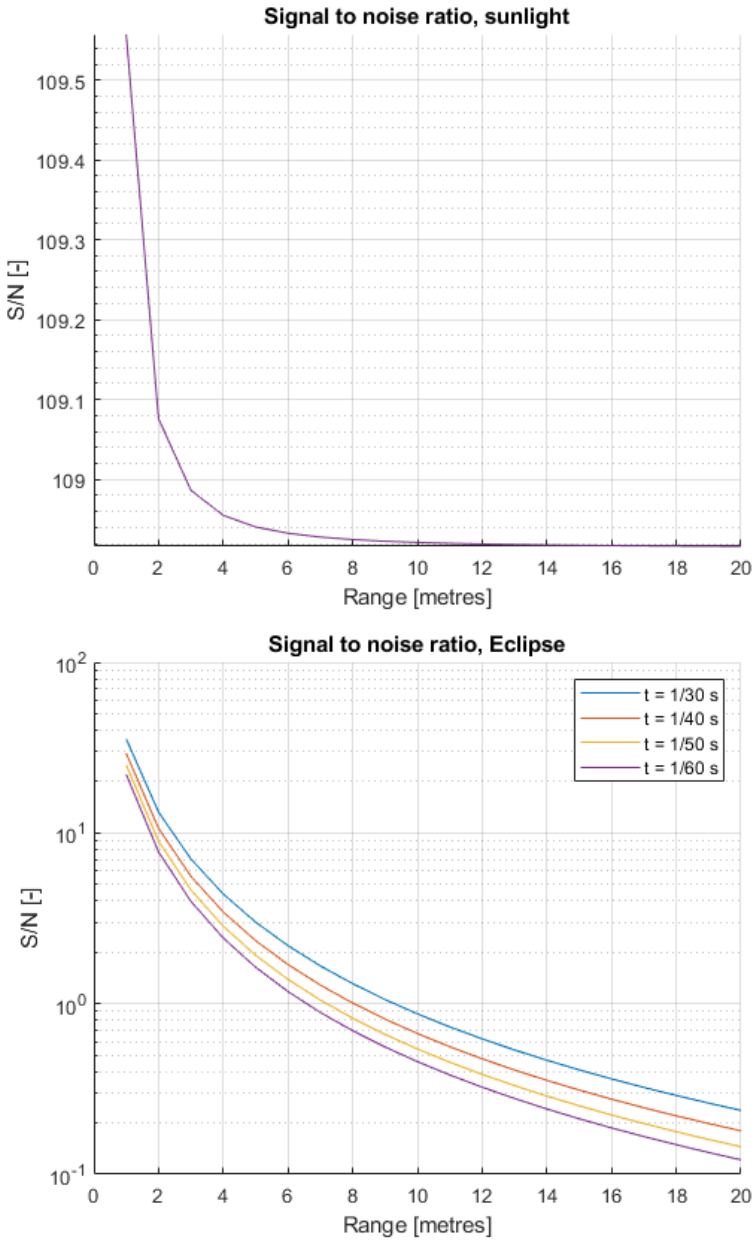


Figure 6.3: Signal to noise ratio in Sun and eclipse.

Plotting the signal to noise ratio for both the Sun illumination case and the eclipse case in Figure 6.3, it can be seen that the signal to noise budget closes easily for the Sun illuminated case (with t = 1 ms). However, if the assumption is made that the integration time cannot be higher than 1/30 seconds, the eclipse case does not close above 8 metres of range. Therefore, one of the design constraints needs to be relaxed in order to make this system feasible. A number of measures could be taken, e.g. reducing the illumination spread, accepting longer exposure times, using more reflective material etc. Relaxing these three parameters (for Figure 6.4 spread from 16 to 8 degrees, exposure times up to 1/50th seconds), allows closing of the illumination budget.

The effect of relaxing constraints is shown in Figure 6.4.

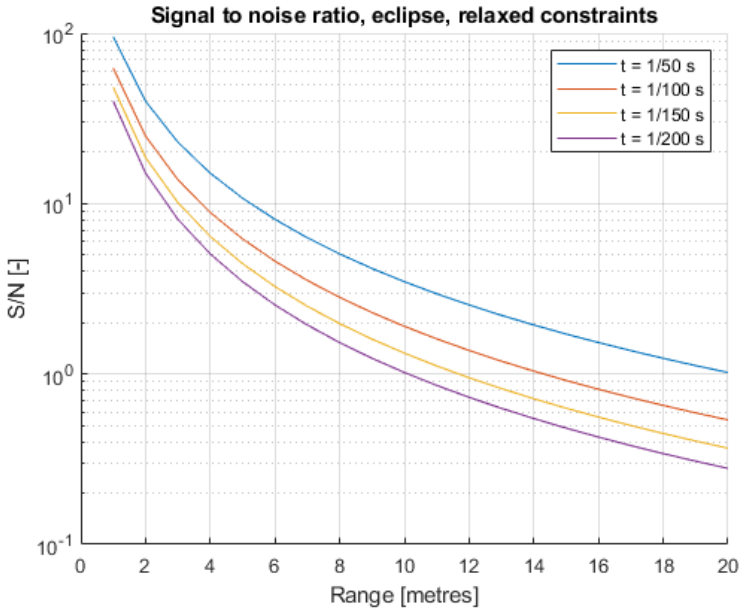


Figure 6.4: Signal to noise ratio for relaxed constraints in eclipse.

The power used by this illumination system can be calculated by assuming a power consumption of 2.1 Watt per LED, corresponding to the datasheet of the component. Multiplying by the number of components and assuming pulsing operations equal to the exposure time of the scene gives the following equation:

$$P_{illumination} = P_{LED} * n_{leds} * t * fps \tag{6.10}$$

If the power consumption is limited at 5 Watts, the maximum fps at t = 1/50 seconds is, for example, 1.2. Naturally, at lower ranges, a number of LED's can be switched off and/or the integration time can be reduced. This calculation shows that the system power consumption can be regulated by accepting a lower number of relative pose estimations per second at range, in addition to changing some of the system parameters noted above that increase the signal to noise ratio of the system.

The calculations are validated by comparing the signal to noise behaviour over distance to a similar calculation reported in [96]. It is concluded that the behaviour is as expected.

6.3. Material Options

The selection of materials for the Mantis Marker is difficult at this stage in the design. Since little research has been done towards the space qualification of many materials, this section is restricted to listing some possible materials, their required specifications and a brief discussion of verification methods for these materials. All materials are considered on their *optical* properties, structural properties are deferred to future work.

For the reflective elements of the Mantis Marker, the system has been modelled so far with a retroreflective element. At range and in eclipse, retroreflectivity is required to a high degree. If the marker is applied in sunlight, a highly diffuse material such as Spectralon may have sufficient reflective properties and has been used in space [97]. With the assumption that retroreflectivity is required, 3M Diamond Grade™, retroreflective tape achieves up to 500 sr^{-1} according to [95]. This is more than the modelled reflectance in the radiometric analysis of the previous section. However, this material has, to the knowledge of the author, not been tested for space applications.

For the black/absorbing elements of the Mantis Marker, the material should be as absorbing as possible to create the highest contrast between the reflective surfaces and all surrounding surfaces. A logical approach would be to use a type of space-grade paint, which allows a wider choice of materials for the encoding elements and base of the marker. A promising design option would be to use the Acktar black, a coating that is extremely absorbing and has been qualified for use in space [98].

For both the reflective surfaces and the absorbing surfaces, the main constraint is the degradation of the optical properties of the materials over the course of 20 years in the space environment. Temperature extremes, atomic oxygen, debris, radiation, charged particles etc. degrade the optical properties of the materials [99]. Most specular materials become more diffuse over time [100]. Furthermore, the materials should prevent outgassing in vacuum. Environmental testing can be performed on ground by appropriately designing verification tests that simulate the equivalent radiation doses, thermal cycling, atomic oxygen flux, vacuum loading etc. of above 20 years in space [101].

6.4. Conclusion

The end-to-end system implementation of the Mantis Marker in a relative navigation system for active debris removal was covered in this chapter.

Two design options for the system layout were presented, a wide and a narrow options. The performance of these designs will be evaluated in chapter 8. In both designs, two dimensions of Mantis Markers were applied to enable relative navigation in the complete operational envelope in terms of range. It was concluded that two opposing faces on two perpendicular axes should be marked in order to prevent approach trajectories that place the Sun behind the target spacecraft.

For the illumination of the target, an infrared LED illumination system was sized. This system attempts to decrease the delta between its application illumination environments: sunlight and eclipse. A bandpass filter is applied to the camera to filter out most of the sunlight.

A radiometric analysis of the end-to-end system was performed. In sunlight, the system signal to noise ratio is sufficient under the made assumptions. However, in eclipse ranges above 6 metres become problematic. It was shown that this can be mitigated by a tradeoff in certain system parameters when operating the system at range and in eclipse.

For the material of the Mantis Marker, two space-grade materials that may be used for the reflective respectively the absorbent elements are Spectralon and Acktar black. Since a retroreflective material may be necessary at range to provide a sufficient signal to noise ratio, 3M Diamond Grade™retroreflective tape was proposed as material option. However, these materials need to be qualified for exposure to the space environment for 20 years. This can be done by environmental testing on ground in purposefully designed experiments.

7

Experimental Setup

To perform verification of the fiducial system design with respect to the system requirements and to validate the system performance with respect to the stakeholder requirements, a series of experiments is performed. Two experiments are performed: a virtual and a physical experiment. The virtual experiment allows for quick iteration, is low-cost and generates a large amount of data. This experiment is aimed at verifying the performance of the fiducial marker system. The physical experiment is aimed at verifying the system functions of the feature extraction software in a real-life scenario and at validating the virtual Blender model.

In section 7.1 the setup of the physical experiment is covered and in section 7.2 the setup of the virtual experiment is covered.

7.1. Test Setup Physical Experiment

The physical experiments for RENDER were performed at the Orbital Robotics and GNC Lab (ORGL) at the ESA European Space Research and Technology Centre (ESTEC).

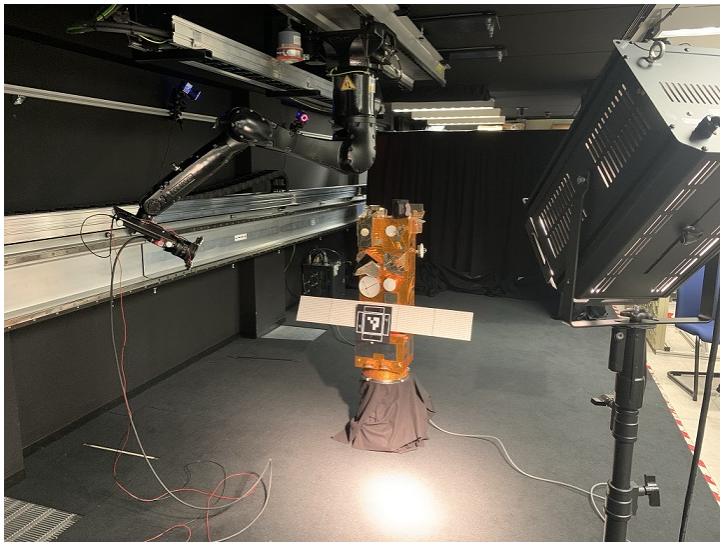


Figure 7.1: Test setup for RENDER in the ORGL. To the left, the robotic arm with the attached camera and illumination system is shown. In the centre, the mockup of Envisat with the applied Mantis Marker is visible. On the right the spotlight is visible. Note that during testing the illumination environment is controlled, not shown here.

Zwick et al. extensively describe the capabilities of the lab in [102]. The lab is intended to support research on GNC and robotic space applications and allows for the testing of, among other things, closed range rendezvous scenarios. To this end, the lab is equipped with one wall-mounted and one ceiling-mounted robotic arm installation called GNC Rendezvous, Approach and Landing Simulator (GRALS). To the end effector of these robotic arms, a camera can be attached. Since the RENDER experiments are focused on the performance of a fiducial target, these robotic arms were used in an open-loop system integration.

The test setup (shown in Figure 7.1) consists of a camera attached to the end effector of the ceiling-mounted robotic arm. To this camera is an illumination system attached which illuminates the target. Different illumination scenarios are tested to verify the system function under eclipse and in sunlight. To this end, a halogen theatre lamp is used. A 20 x 20 centimetres prototype of the Mantis Marker was manufactured using 3D printing. A mock-up of Envisat was used to mimic typical shapes present around a fiducial target on a spacecraft. All results were recorded using Simulink.

7.1.1. Camera

The detector that was used for the experiments is the Prosilica GT4096 NIR from Allied Vision. This is a monochrome camera that has the capability to capture a spectrum of light including near-infrared. Its key specifications are listed in Table 7.1.

Parameter	**Value**	**Unit**
Resolution	4096 x 4096	Pixels
Sensor type	CMOS	-
Sensor size	Type APS-H	-
Pixel size	4.5 x 4.5	μm
Temporal dark noise	28.2	electrons
Max. frame rate at full resolution	7.18	fps

Table 7.1: Prosilica GT4096 NIR specifications [94]

Attached to this detector is the Canon EF 24-70mm f/4L IS USM lens. This is a zoom lens with variable focal length and adaptable focus. The FOV of the entire camera (detector and lens) is calculated via the following equations:

$$A_d = Res_v * Res_h * Ps_x * Ps_y \tag{7.1}$$

In Equation 7.1, A_d is the detector area, Res_v and Res_h are respectfully the vertical and horizontal resolution, Ps_x and Ps_y respectfully are the horizontal and vertical pixel dimensions. Since both the detector and pixels have equal dimensions on both sides, the square root of A_d gives both the horizontal and vertical dimension of the detector in metres:

$$Dd_v = Dd_h = \sqrt{A_d} \tag{7.2}$$

Using $Res_v = Res_h = 4096$ and $Ps_x = Ps_y = 4.5 * 10^{-6} m$, Dd_v and Dd_h are calculated to be 18.43 mm. It is then possible to calculate the FOV given a known focal length:

$$FOV = 2 * tan^{-1}\left(\frac{Dd_v}{2f}\right) \tag{7.3}$$

Where f is the focal length in metres. The focal length depends on the setting for per particular experimental run. For $f = 24$ mm, the FOV is 42.01 deg.

7.1.2. Illumination System and Bandpass Filter

A custom illumination system was made to illuminate the target (shown in Figure 7.2). This illumination system consists of 3 clusters of SFH 4783 infrared LEDs with a half angle of 12 degrees. The LEDs are powered by a current limited power supply and mounted using a custom-designed and 3D printed camera mount. To this mount, active cooling is applied using fans. The LED arrays were mounted in parallel and powered by approximately 8.8V and 2A.

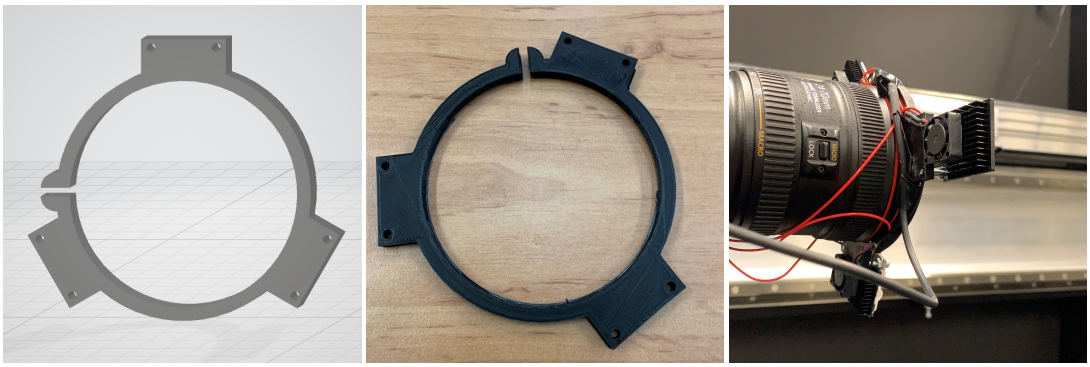


Figure 7.2: Led mount. From left to right, drawing of mount, printed mount, integrated mount with active cooling.

In addition to the illumination system, a bandpass filter is applied to the camera to simulate the end-to-end implementation as proposed in chapter 6. The bandpass filter implemented is the Midopt LP715-77 longpass filter. This bandpass filters all light with a wavelength below 715 nm. The setup is shown in Figure 7.3.

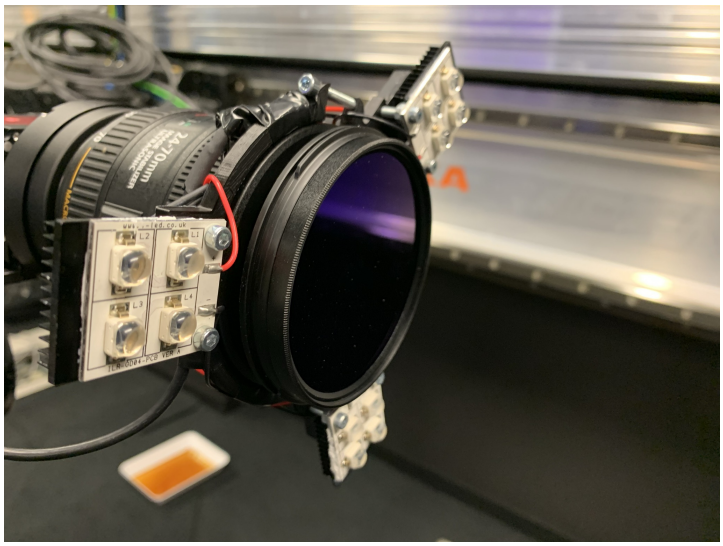


Figure 7.3: Bandpass filter applied to camera.

The Sun was simulated using a Halogen spotlight, which is assumed to provide illumination in a sufficient degree of similarity in terms of spectrum with regards to the Sun. For every run of the physical experiment, the illumination was measured using a light meter. Since the spotlight spectrum was not calibrated, it was not used to validate the system illumination sizing performed in chapter 6.

7.1.3. Mantis Marker Prototype and Target Setup

The Mantis Marker prototype was made using a 3D printer for the printing of the encoding elements, infrared retroreflective tape for the reflective elements and a very absorbing acrylic paint called Black 3.0 for the black elements. The production process is shown in Figure 7.4. A square piece of plastic was used as a base. The di-

mensions of the prototype are driven by the resolution of the 3D printer, which limited the encoding element size to 10 x 2 x 1 cm. Accordingly, the entire marker is 20 x 20 centimetres. Visible in Figure 7.1, the marker is mounted on a satellite model that is not representative of the relative scale of the marker to the spacecraft. However, since the shapes around the marker are representative of the shapes present on a real spacecraft, this relative size is of lesser concern.

Figure 7.4: Mantis Marker prototype. From left to right, 3D printed encoding elements, painting of the prototype, finished Mantis Marker.

7.1.4. Trajectories

Two experimental runs were performed, one with simulated sunlight and one in eclipse. For both runs, a KUKA trajectory was used that included 4 passes of the fiducial marker at different heights to simulate a range of combined pitch and yaw rotations. Each pass is performed two-way, so in total 8 one-way passes are made for a single run. The trajectories are shown in Figure 7.5.

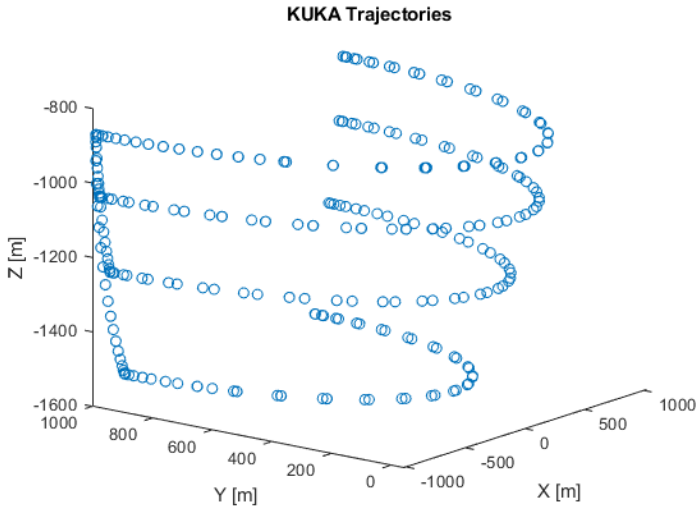


Figure 7.5: Plot of programmed KUKA arm trajectories.

7.1.5. Image Scaling

Since the camera records data at high resolution, the marker is relatively large and the distance between camera and marker is small, the internal planar fiducial causes no failure of the pose solution for the taken data (this is shown in chapter 8). However, the Blender model needs to be validated using this data. In order to provide a comparison of data, the image needs to be scaled down to be representative of the Blender data collected.

To do so is relatively straightforward: first, the image is scaled down and padding is added by repeating the outer pixel values to have the fiducial be of comparable size relative to the images generated by the Blender model. Next, the resolution is strongly reduced. Finally, the camera intrinsics are scaled according to the new image resolution and virtual pixel size. The output is shown in Figure 7.6.

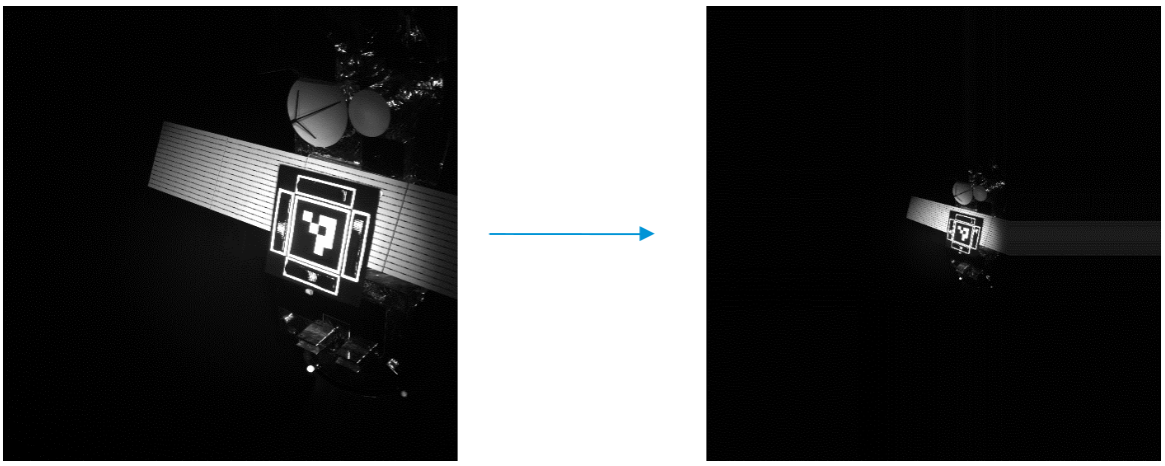


Figure 7.6: Image scaling to enable Blender model validation.

7.1.6. Camera Intrinsic Calibration

To estimate the pose of a fiducial, the intrinsic parameters of the camera need to be known. The intrinsics can be found using an approach proposed by Zhang in [103]. This approach can be summarised as taking a range of images at different relative poses using a "chessboard" i.e. a plane with a number of alternating black and white squares, shown in Figure 7.7. Using an initial guess of the intrinsic parameters found by finding a homography, the world coordinates of the corner points and the image coordinates of the same corner points found by a feature extraction algorithm, the intrinsics are found by iterating until the reprojection error reaches a certain threshold. All code was implemented using a range of OpenCV functions and can be found in Appendix A.

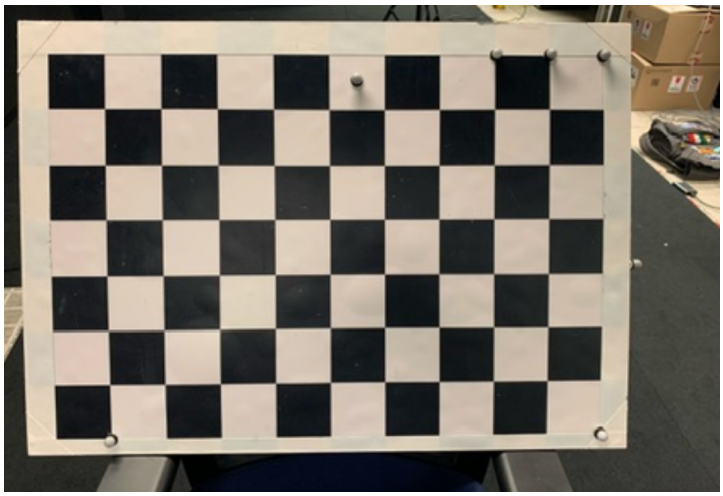


Figure 7.7: Calibration target.

7.2. Test Setup Virtual Experiment

For the virtual tests of the design, a Blender model was used. Blender is a free and open-source "3D creation software suite"[1]. It enables the rendering of images using ray tracing, thus enabling the testing of fiducials to some degree of realism. A screenshot of the setup in the Blender software is shown in Figure 7.8.

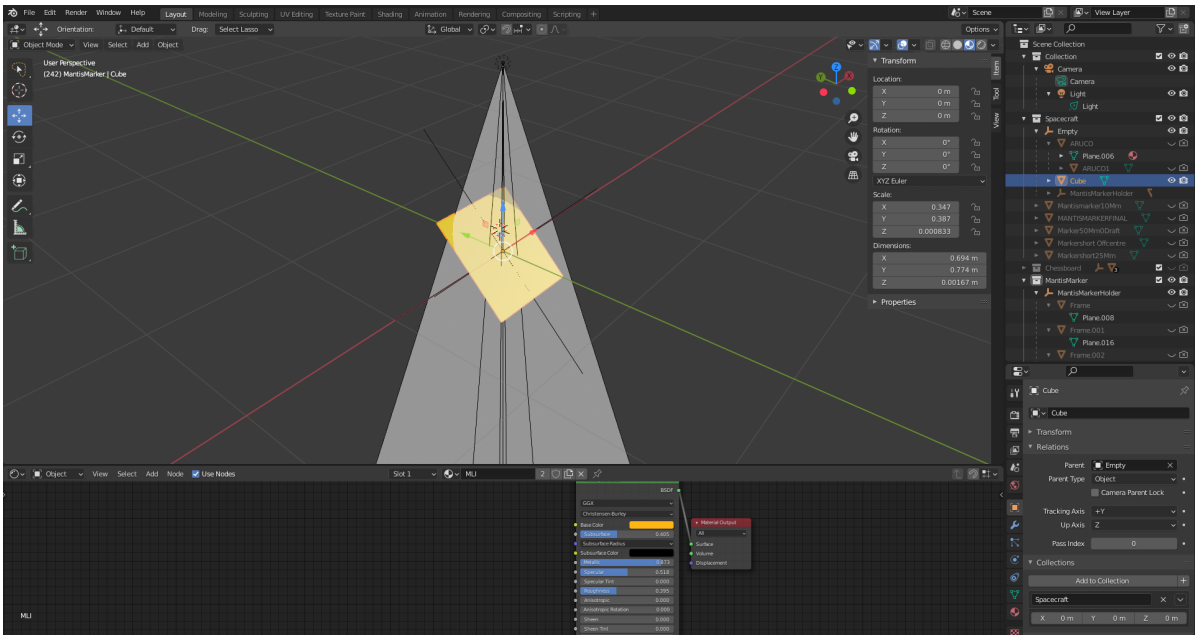


Figure 7.8: Screenshot of the Blender experimental setup for RENDER.

For each test in the design process, the Solidworks design of the encoding element at that iteration was loaded into a mock-up of the complete fiducial (including an Aruco marker). The appropriate camera intrinsic parameters, scaling and staring positions of all elements and illumination conditions were set. Due to the required lens centred illumination, a virtual spotlight was programmed to coincide with the camera principal point and imaging direction.

Blender is scriptable in the Python programming language, and a script was implemented that automatically takes a range of images for set fiducial orientations and relative positions. For each run, a calibration set was taken with a virtual chessboard, such that the written calibration software could also be tested and to ensure consistency in the camera intrinsics.

While the Blender model is validated using experimental data in chapter 8, some limitations of the model were determined beforehand. Firstly, it is assumed that the illumination system and the propagation of light are not accurately modelled by Blender. Accurate light propagation is arguably difficult to model virtually. However, since no optical elements such as lenses have been used in the Mantis Marker, it can be reasonably assumed that the basic propagation of light through slits can be modelled accurately. Secondly, the reflective properties of both the used marker material and the surrounding spacecraft are not modelled to a realistic level.

The application of the Blender model is thus the testing of the layout of the Mantis Marker and the feature extraction software as well as the overall performance of the system with the absence of adverse illumination conditions. These conditions need to be tested separately in a physical experiment.

To benchmark the Mantis Marker, an Aruco marker is used since it is a representative and often used planar fiducial that is readily available in OpenCV. In addition, the Aruco corners can be refined using the Apriltag 2 approach detailed in [104]. This increases the performance of the planar fiducial as well as the Mantis Marker

[1] The Blender software can be found at: https://www.blender.org/

and is therefore a more accurate representation of the state of the art performance.

The Blender model camera has a resolution of 720 x 480 pixels. The camera has an FOV of 7.4 degrees horizontal and the simulated detector dimensions are 4.512 x 4.988 mm.

7.3. Conclusion

In this chapter, both the physical and the virtual experimental setups were covered. The physical experiment was performed at the ORGL and included a robotic arm, custom illumination and a 3D printed fiducial. The experiment was intended to verify the functions of the feature extraction software in a real-life scenario and to validate the virtual Blender model. To achieve the latter, a scaling method was used on the taken images.

The virtual experiment allowed for quick iteration and generating large amounts of data. The data was generated using Blender. This experiment enables the simulation of the performance of the system with the absence of adverse illumination conditions. The model was scripted using the Python language.

The results of both experiments are presented in the next chapter. Moreover, the validation of the Blender model by the physical experiment data is presented.

8

Results

This chapter will present the results from the experiments presented in the previous chapter. In addition, the results from the Monte Carlo simulation of the end-to-end system design options as presented in chapter 6 are shown. Furthermore, the results from a Monte Carlo simulation of an identical fiducial layout *without* encoding elements is presented to show the increase of performance due to the addition of virtual points.

In section 8.1 the results of the virtual experiment are presented. Next, in section 8.2 the results of the physical experiment are shown. Following these experiments, the recorded failure modes of the system are analysed and mitigation strategies are proposed in section 8.3. Subsequently, the results from the Monte Carlo simulation are presented in section 8.4. Finally, system verification and validation is performed in section 8.5.

8.1. Virtual Experiment Results

The virtual experiments performed using the Blender model described in section 7.2 are intended to verify the system performance with respect to the prevention of pose ambiguity and to qualitatively benchmark the performance with respect to existing planar fiducials by comparing to an Aruco marker of equal dimensions. A series of experiments have been performed, with the following characteristics:

Test No.	Marker Size [m]	Marker	Distance [m]	Rotation
1	0.025	Mantis Marker	3.3	Pitch
2	0.025	Aruco	3.3	Pitch
3	0.025	Mantis Marker	2.3	Pitch
4	0.025	Aruco	2.3	Pitch
5	0.025	Mantis Marker	1.3	Pitch
6	0.025	Aruco	1.3	Pitch
7	0.025	Mantis Marker	2.3	Pitch, yaw
8	0.025	Aruco	2.3	Pitch, yaw
9	0.025	Improved Mantis Marker	1.3	Pitch

Table 8.1: Test description virtual experiments

These parameters have been chosen such that they are comparable to the test parameters used in [60]. A data sample at each distance is shown in Figure 8.1. The marker is rotated around the shown rotation axis. The rotational range for the pitch axis is between -45 and 45 degrees and incremented by 0.1 degrees per scene. For the experiments with a combined rotation around the pitch and yaw axes, the rotation around both axes was the same for each scene and between -35 and 35 degrees (i.e. [-35,-35],[-34,-34] ... [35,35]), since this is the operational envelope of the encoding elements as implemented in the tested Mantis Marker.

Figure 8.1: Sample of test images at ranges of 1.3, 2.3 and 3.3 metres.

First, the results from run 1 and 2 are compared. At a range of 3.3 metres, the Mantis Marker greatly outperforms the Aruco marker in terms of pose stability as well as accuracy under frontal observations, shown in Figure 8.2. Interestingly, when looking at the range solution for both markers, it appears that the Aruco solution is more stable than that of the Mantis Marker. This is visible in Figure 8.3.

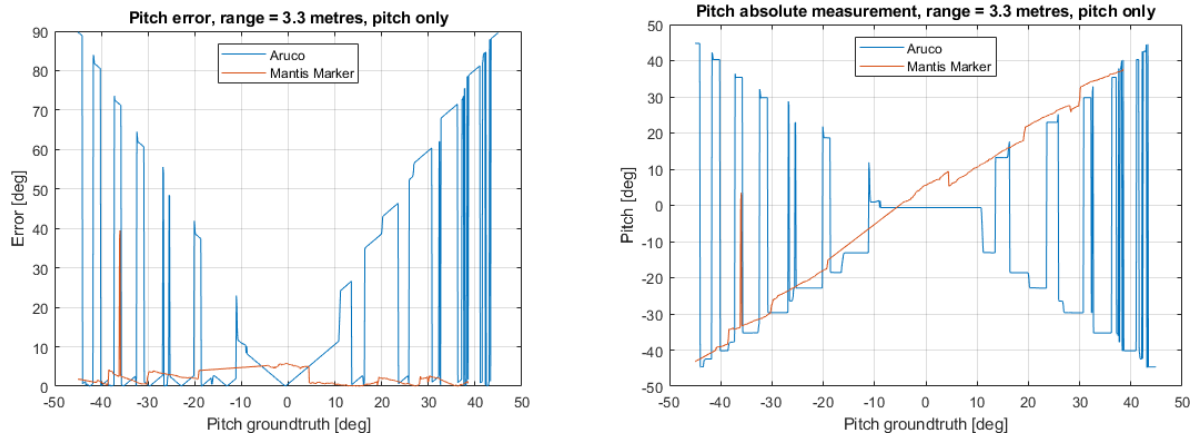


Figure 8.2: Results for test 1 and test 2 compared in attitude measurements.

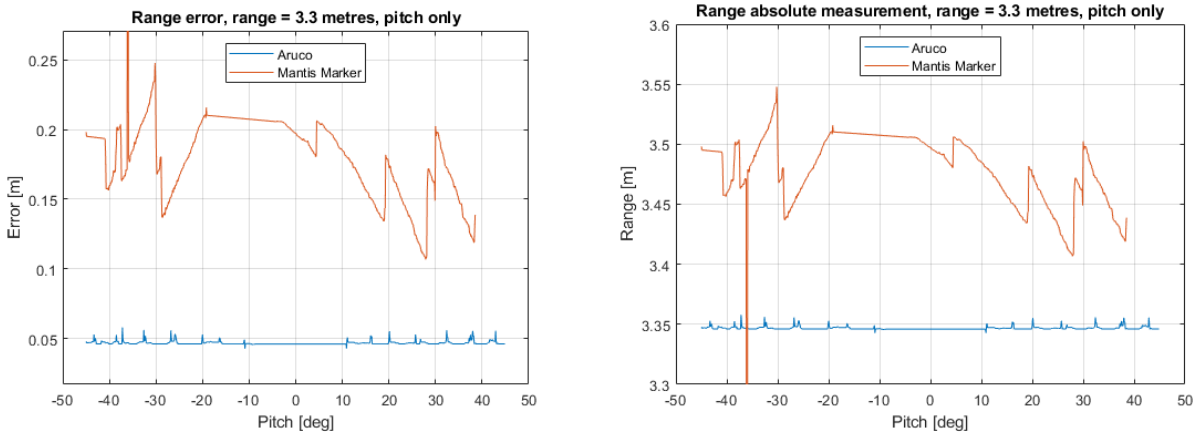


Figure 8.3: Results for test 1 and test 2 compared in range measurements.

The stability of the Aruco marker with respect to the Mantis Marker in terms of range is due to the reduced size of the Aruco inside the Mantis Marker in its current design. This can be solved by a design improvement, namely using the corners of the complete marker as feature points in the PnP solution, therefore effectively increasing the Mantis Marker feature points to the same dimensions as those of the Aruco. The only limitation in doing so is the required binary pattern size for unique identification, which, with the limited number of markers necessary on a spacecraft, can be greatly reduced with respect to the standard Aruco library.

A second important observation is the lack of pose solutions for the Mantis Marker at this range for measurements above a positive yaw of 38 degrees. Notably, the Aruco solutions for this pitch range also oscillate severely. It appears that the Aruco marker used in this particular setup has an asymmetry in the detectability of the encoding bit pattern. At the very limit of the experimental domain in terms of range and attitude this, together with the reduced size of the internal Aruco of the Mantis Marker with respect to the benchmark Aruco, causes the feature extraction software to be unable to find the marker.

For run 3 and 4, the pitch measurements are plotted in Figure 8.4. The Aruco marker also suffers from pose ambiguity as well as reduced precision in frontal observations at this range. The Mantis Marker has a small number of pose jumps, where the error is practically equivalent to the error of the Aruco marker in terms of pitch.

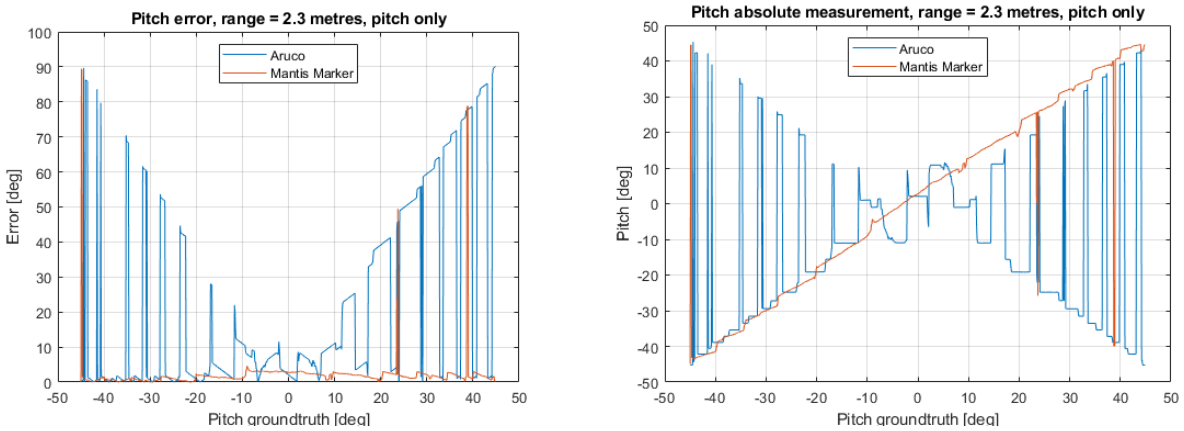


Figure 8.4: Results for test 3 and test 4 compared in attitude measurements.

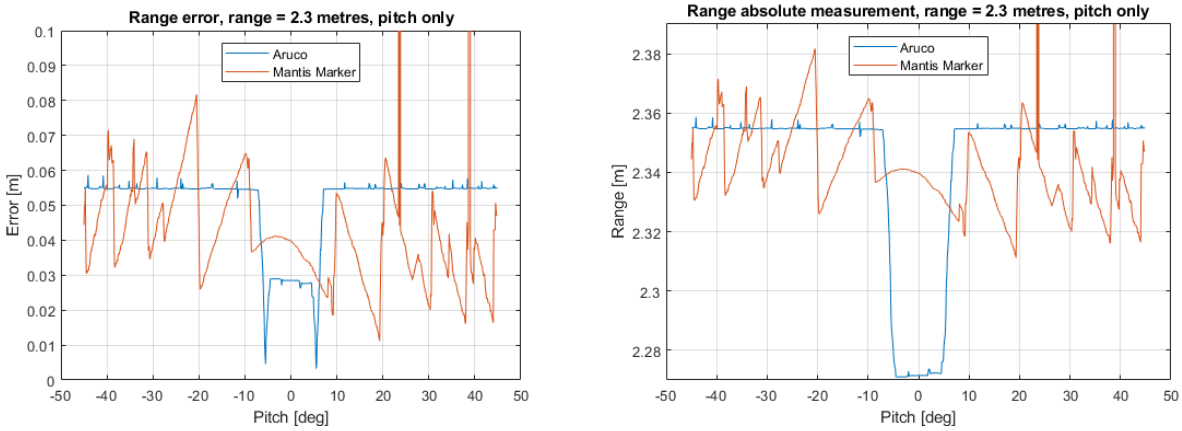


Figure 8.5: Results for test 3 and test 4 compared in range measurements.

In Figure 8.5, it is shown that the severity of the outliers for the Mantis Marker in terms of range is much higher than the pose outliers of the Aruco marker. The increased severity of the outliers is due to the addition of a fifth point in the pose solution, forcing the pose solver to a local minimum far from the actual solution. Due to the severity of these outliers, it could be argued that these are in fact easier to filter in an end-to-end system than the less severe Aruco marker solution outliers. The left-most outlier of the Mantis Marker visible in the pitch error in Figure 8.2 is not visible in the range solution. This is likely due to the fact that the pipeline reverts to the solution of the internal Aruco marker, in the case that the feature extraction algorithm of the Mantis Marker pipeline is unable to find a complete virtual point.

When comparing the results of test 5 and 6, between the Aruco and Mantis Marker at a range of 1.3 metres, it became apparent that the Mantis Marker solutions were unstable. Since the results were expected to be in line with those shown in Figure 8.4, i.e. strong robustness of the attitude solution, this is surprising. However, further inspection and results from the physical experiment proved that this was due to a failure mode of the feature extraction software that caused the corners of the internal Aruco of the Mantis Marker to be identified at the wrong location. The cause and mitigation of this failure mode are covered in section 8.3. To provide a comparison of the Aruco marker and the Mantis Marker for this distance, the design was improved to prevent this failure mode from occurring and the experiment was repeated. A render of the design is shown in Figure 8.6, where it can be seen that the white space around the internal Aruco marker is made broader.

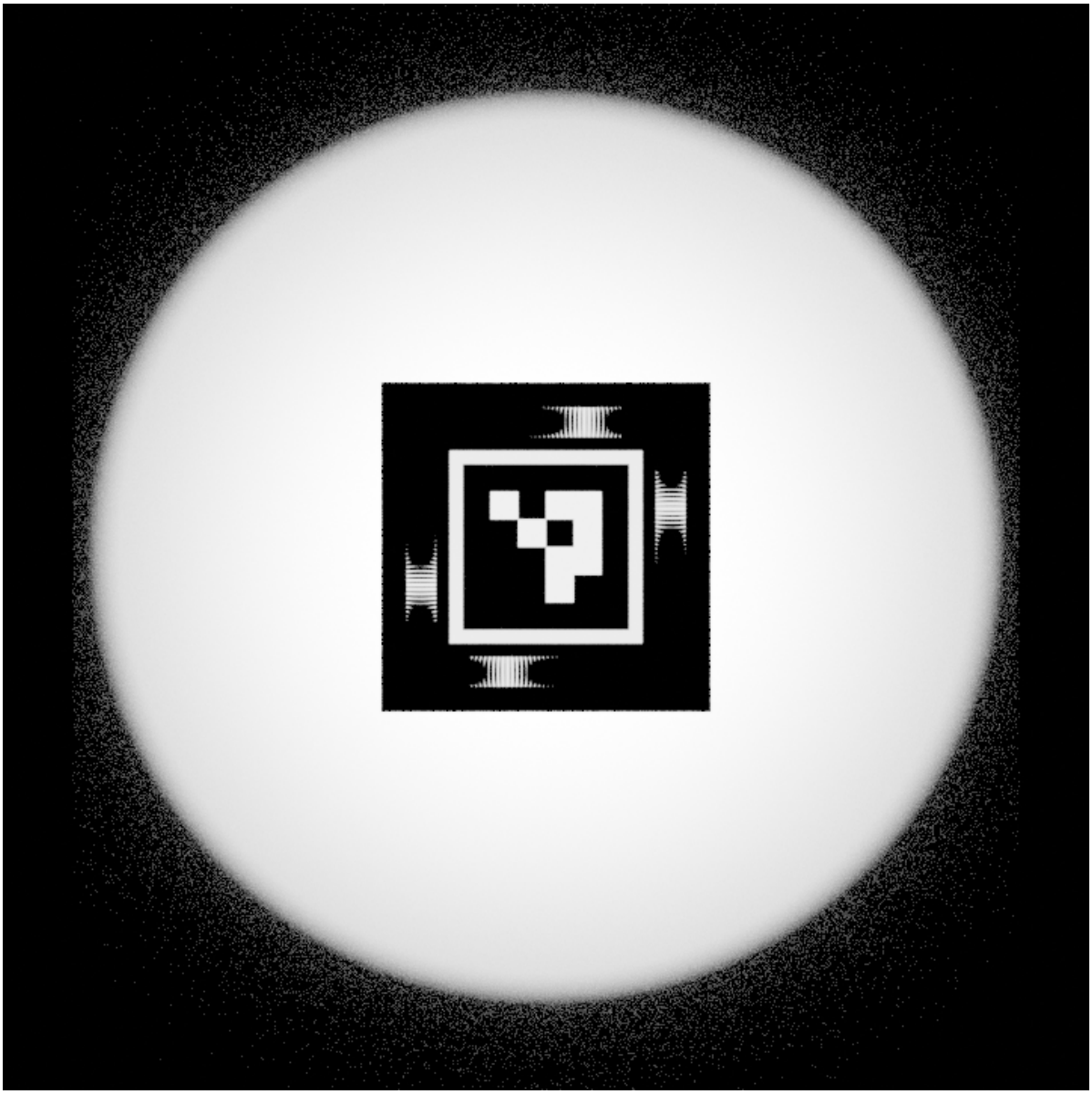


Figure 8.6: Tested improved design including broader white space.

The Aruco solution at this range is much improved in stability with respect to the measurement distance of 2.3 metres. While the Mantis Marker is still more stable for pitch measurements, a bias of around 0.045 metres is present for the range measurement, similar to the measurements of test 1. However, the Aruco bias has been greatly reduced with the smaller range, outperforming the Mantis Marker. This is again due to the relatively smaller size of the internal Aruco in the Mantis Marker. The pose solver is able to find a more accurate local minimum for the pose for a larger planar element, which is expected.

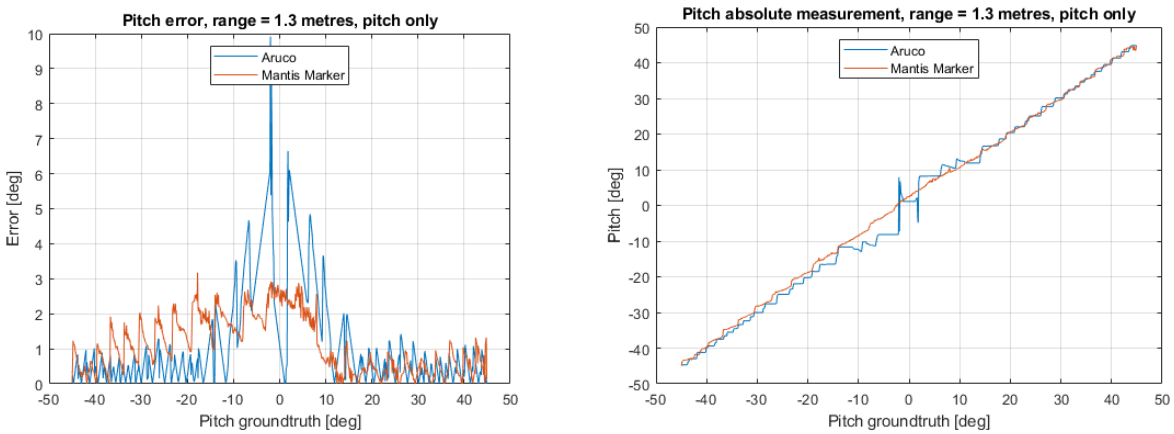


Figure 8.7: Results for test 6 and test 9 compared in attitude measurements.

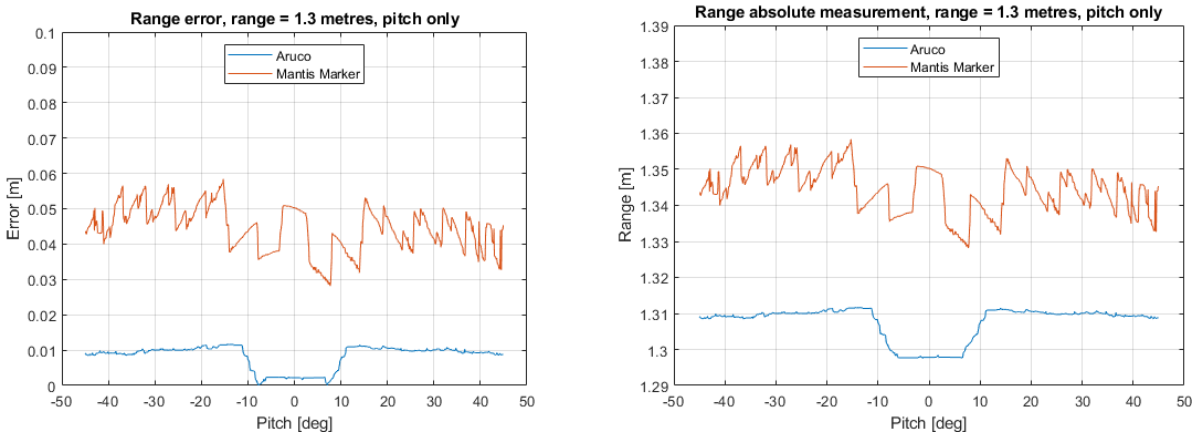


Figure 8.8: Results for test 6 and test 9 compared in range measurements.

Somewhat surprisingly, there appears to be an asymmetry in the pitch error for the negative values of the pitch ground truth with respect to the positive values. This is clearly visible in Figure 8.7 and to a lesser extent visible in Figure 8.4. This is unexpected since the experiment is performed in idealised conditions. Both the illumination environment as well as the Mantis Marker are designed to be entirely symmetrical. Therefore, this must be a function of the current design of the encoding element. A further inspection of the virtual Mantis Marker showed an alignment error of 0.001 metres of the left encoding element. This explains the bias that is present up to a pitch angle of around 8 degrees, where the overlap between the encoding elements ends and only the right encoding element is used for the pose estimation. This bias is more clear at the close range measurements since the delta in virtual point alignment is more significant due to the increased angular difference. In future work, such a bias should be caught by appropriate calibration procedures.

To test the influence of combined rotations, an experiment with both pitch and yaw rotation was performed, shown in Figure 8.9 and Figure 8.10. While the Mantis Marker shows no change in performance in pitch determination, the Aruco marker performance improves with respect to the measurements at 2.3 metres under only pitch rotation. This is due to the increased perspective effects under combined rotation, reducing the occurrence of pose ambiguity. For both markers, the precision and accuracy of the range estimation is reduced. With the increased perspective effects, stability of the solution for the scale.

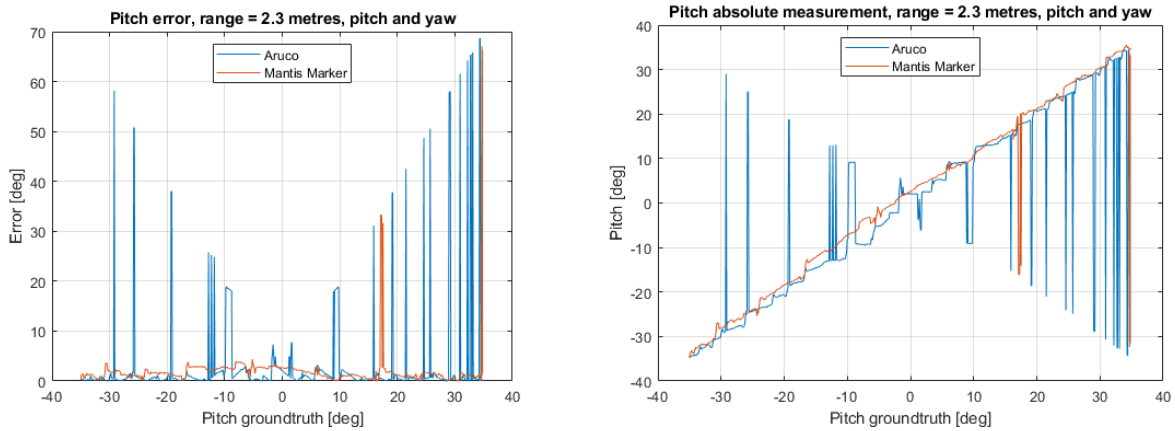


Figure 8.9: Results for test 7 and test 8 compared in attitude measurements.

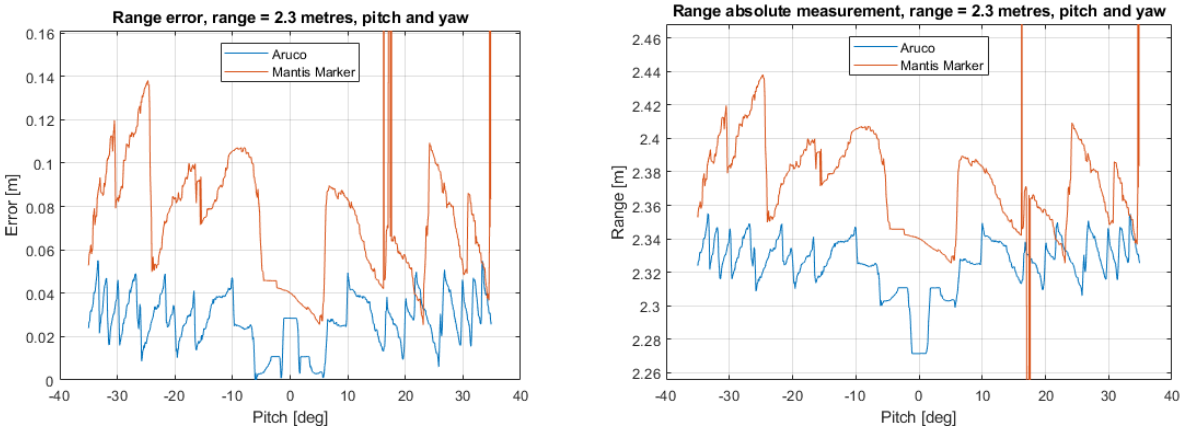


Figure 8.10: Results for test 7 and test 8 compared in range measurements.

With the combination of these results, it can be concluded that the Mantis Marker greatly improves the attitude estimation of planar fiducials. It prevents pose ambiguity as well as precision loss under frontal observations and outperforms the Aruco marker in attitude estimation. While the current implementation shows worse performance for range estimation, this could be mitigated by including the Mantis Marker corners as features in the PnP solution. At larger ranges, the relative performance increase of the Mantis Marker with respect to the Aruco is larger due to the increased likelihood of pose ambiguity. Under increased perspective projection, such as relative poses under combined rotations, the relative performance increase is reduced but still apparent.

8.2. Physical Experiment Results

The goal of the performed physical experiments is to verify the system functions in a real-life scenario and to validate the Blender virtual model. Two experimental runs are performed, one in eclipse and one in simulated sunlight from an incident angle of approximately 45 degrees. The performed robotic arm trajectories are described in subsection 7.1.4. For both runs, the failure modes of the system are recorded and analysed.

To verify the system functions, the yaw solution for the Mantis Marker is plotted against the yaw solution of the internal Aruco marker. While this is not a fair comparison of performance since the internal Aruco is much smaller than that of the entire marker size, it serves to verify the functioning of the software in a real-life scenario and the response of the system to illumination.

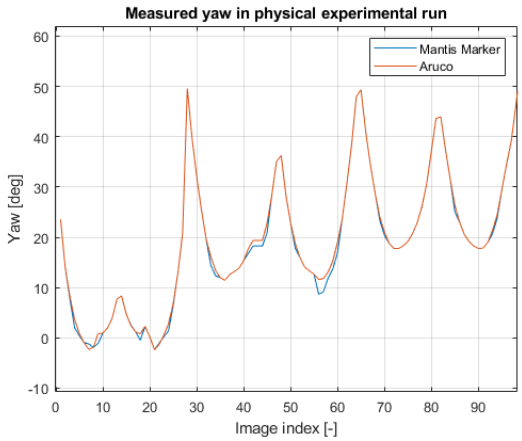


Figure 8.11: Yaw solution for the physical verification of the Mantis Marker versus the internal Aruco marker.

In Figure 8.11, the yaw solution of the experiment is presented. The 3 passes of the robot arm are visible. The forward and backward motion of the robotic arm causes the 3 distinct "W" shapes visible in the plot. The solution of both the Aruco and the Mantis Marker is nearly identical, verifying the software of the Mantis Marker on real images. Some very minor instability is visible for the Mantis Marker on more frontal observations, this is due to the same failure mode as was visible in the Blender model for test 6: faulty corner detection.

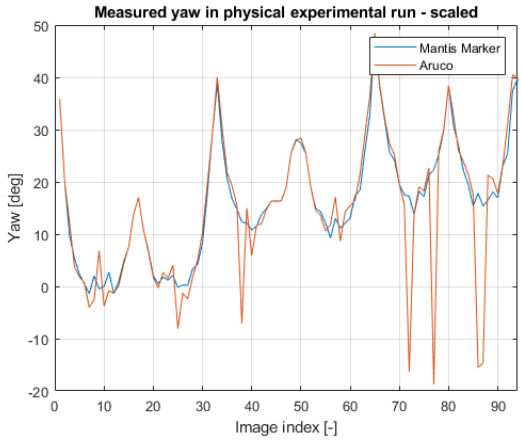


Figure 8.12: Yaw solution for the physical verification of the Mantis Marker versus the internal Aruco marker, with scaled image to mimic same relative marker size as Blender experiment.

Due to the high resolution of the camera and the low range, no pose ambiguity is present in Figure 8.11. To incite pose ambiguity and loss of precision in frontal observations, the resolution of the image and the relative size of the marker with respect to the image size needs to be artificially reduced. This process is explained in subsection 7.1.5. The result of scaling the images and recording the yaw solutions on these images is shown Figure 8.12. Here, the Aruco shows a loss of precision under frontal observations as well as pose ambiguities while the Mantis Marker does not. This is similar to the behaviour of these markers as recorded by virtual experiments, validating the Blender model.

For the response of the system under sunlight, an additional run was recorded. Under the tested configuration, the system was only able to identify the blob of the Mantis Marker encoding element when the robotic arm positioned the camera such that the simulated sunlight was parallel to the camera direction. The cause

for this is clarified by the images shown in Figure 8.13. Due to the tuning of the exposure time to fit the illumination environment, the brightness of the blob under sunlight is greatly reduced. While still clearly visible to the human eye, the current tuning of the feature extraction algorithm makes it unable to find the blob in these conditions. However, it is reasonable to assume that if the blob location is this clear to the human eye, some minor tuning of the sensitivity of the feature extraction algorithm to the brightness of the blob should resolve this issue.

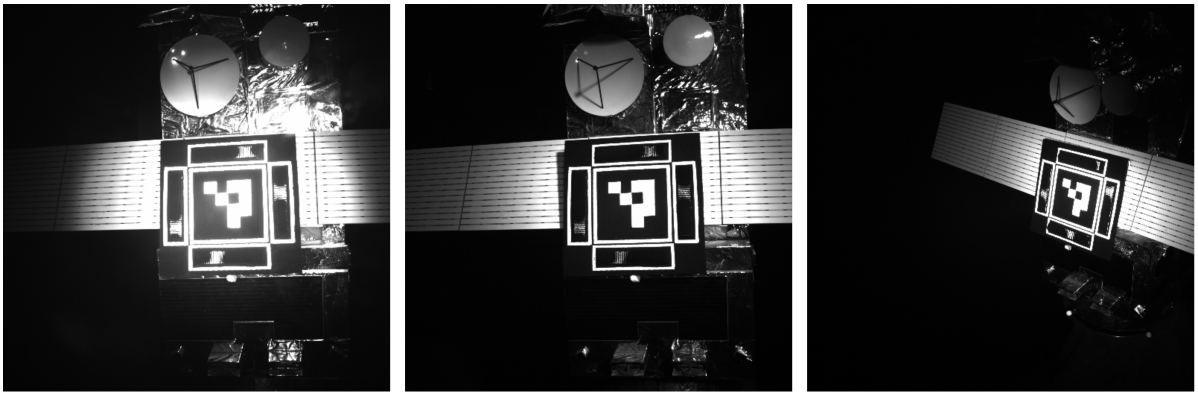


Figure 8.13: Result for system verification under sunlight. From left to right is an image that was taken in eclipse for which the system is able to find the blob in the Mantis Marker encoding element, an image taken in full sunlight for which the system is not able to find the blob, an image for taken in full sunlight (parallel to the camera direction) for which the system is able to find the blob.

8.3. Failure Modes

A number of failure modes were observed in the experimental data. While rare, these pose a risk to any relative navigation system that applies the Mantis Marker. To improve the accuracy of the pose estimation in future work and assess the risks these failure modes pose in an application, these failure modes, their consequences and if possible mitigation methods are analysed here.

The following failure modes are analysed:

1. Loss of Aruco
2. False positive for Aruco identification
3. Complete loss of Mantis Marker
4. Faulty corner detection
5. Inability to find blob

For the analysis of these failure modes and their associated risks, a qualitative risk assessment is made. Three levels of consequence and three levels of likelihood are defined in Table 8.2 and Table 8.3, respectively. This risk assessment is for the marker system as it was tested: a singular marker, without a navigation filter.

Level	Consequence for relative navigation using single marker
3	Complete loss of pose estimation
2	Major loss of accuracy of pose estimation
1	Minor reduction of accuracy of pose estimation

Table 8.2: Consequence levels for qualitative risk assessment of Mantis Marker fiducial system

The consequence and risk levels noted below are assumed to be *pre-mitigation* and are summarised in Figure 8.14.

Level	Likelihood of occurrence
3	Less than 1:10 occurrence
2	Less than 1:40 occurrence
1	Less than 1:100 occurrence

Table 8.3: Likelihood levels for qualitative risk assessment of Mantis Marker fiducial system

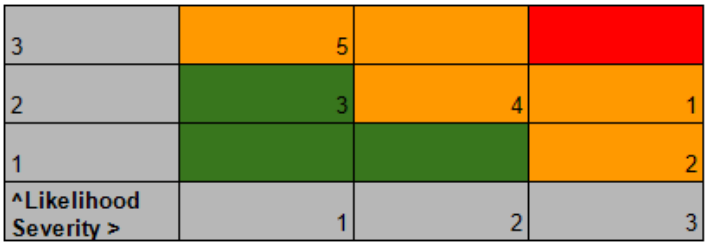


Figure 8.14: Risk assessment failure modes individual Mantis Marker pre-mitigation

Loss of Aruco - *Consequence:* 3 - *Likelihood:* 2 - *Mitigation:* Redundancy
Loss of Aruco can occur in two stages of the feature extraction algorithm for the Aruco marker. The first is the initial feature extraction. If no square shape is detected that satisfies the posed conditions on shape (due to adverse illumination, occlusion etc.), then no candidate for identification is returned and consequently no Aruco is found. Conversely, when one or more shapes is found but the bit pattern attributed to the found marker candidates does not satisfy the error correction conditions, then no candidate is accepted and likewise no Aruco is found. The consequences of this failure mode are a complete loss of pose estimation for that particular marker. However, with the right placement of the marker and proper illumination conditions, this failure mode is unlikely to occur, and if it occurs this will likely only be for a small number of frames (until the causing conditions change due to e.g. a different relative pose or illumination). This risk can be mitigated by including multiple fiducials on the target spacecraft, providing redundancy in the case of single marker failure.

False positive for Aruco identification - *Consequence:* 3 - *Likelihood:* 1 - *Mitigation:* Algorithmic verification and/or navigation filter
In rare cases, the Aruco identification algorithm may recognise a square feature on the spacecraft or its surroundings as a marker. This has been observed a small number of times in experimental data. The consequence is a different number than expected of observed markers. Usually the false positive is a large outlier. The mitigation for this failure mode is thus straightforward: using simple outlier detection and/or navigation filtering these can be removed.

Complete loss of Mantis Marker - *Consequence:* 1 - *Likelihood:* 2 - *Mitigation:* Improved feature detection and/or redundancy
At times, the Mantis Marker feature extraction algorithm cannot find the Mantis Marker. This can be due to a number of reasons. Since the feature extraction relies on the proper detection and localisation of known Aruco corners, a failure of the corner detection usually causes the Mantis Marker detection to fail also. Furthermore, due to occlusion or adverse illumination, the Mantis Marker may not be distinguishable in some images. The consequence is minor: the navigation system can use the Aruco marker to perform pose estimation, resulting in a minor loss of accuracy. Mitigation consists of implementing a feature extraction algorithm that detects the edges of the Mantis Marker (i.e. using Canny edge detection [105]), in conjunction with visible edges around the Mantis Marker. A complimentary mitigation strategy is again redundancy in the applied markers.

Faulty corner detection - *Consequence:* 2 - *Likelihood:* 2 - *Mitigation:* Improved marker design
This failure mode was observed to be most prevalent at close range and frontal observations. It is present in both the Blender experimental data and the physical experimental data. In Figure 8.15, the failure mode is shown. The green border around the Aruco marker should be located around the inner corner of the marker, not at the outside corners. The cause for this failure mode was identified to be an ambiguous corner refinement.

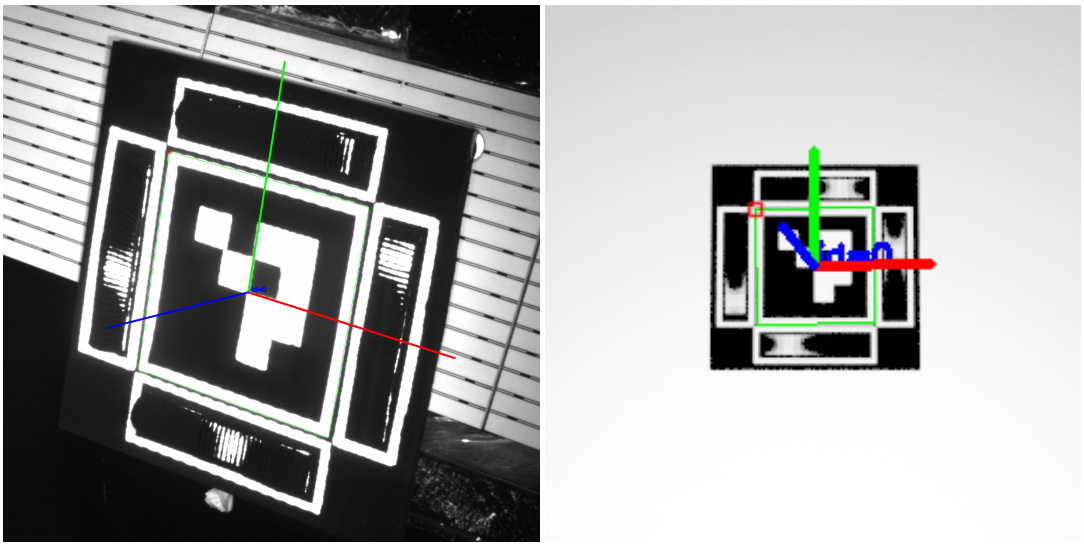


Figure 8.15: Faulty corner detection of Aruco marker in physical and Blender experimental data. A green border can be seen around the outside edges of the white border around the Aruco marker. This border should be present at the inner edges.

Ambiguous corner refinement happens when the feature extraction method comes across an ambiguous corner detection. This may happen due to illumination conditions, which is why in this experimental setup it was most prevalent in frontal observation where strong illumination caused the correct corner candidates to be less pronounced. Furthermore, this may happen due to marker design: in the tested prototype, a small gap was present between the border around the Aruco marker and the border around the Mantis Marker. At close range, this can cause ambiguous corner detection due to the search window of the corner detection algorithm including two corners, as depicted in Figure 8.16. This effect was also described in [106]. Normally, this becomes more prevalent at larger range where the search window is proportionally larger than the edges of the marker. However, in the case of the used test setup, the gap was too small to be distinguished by the feature extraction pipeline at the medium and long range.

Faulty corner detection has a consequence of a major loss of accuracy of the pose estimation. Both range and orientation can become distorted and the failure mode causes the Mantis Marker to be undetectable in nearly all cases since the relative location is faulty. The mitigation of this failure mode is straightforward: if an Aruco is used as primary marker in future work, this marker should include a wider border around its faces. The recommended design changes are further covered in chapter 9.

Inability to find blob - *Consequence:* 1 - *Likelihood:* 3 - *Mitigation:* Redundancy and/or increased chaser capabilities and/or algorithmic improvement
Relatively often, one or more blobs cannot be found by the Mantis Marker feature extraction algorithm. Apart from an angle of view that exceeds the FOV of that particular marker, this may occur due to imperfections in the feature extraction algorithm. The consequence is minor to non-existent: as long as two perpendicular blobs can be found, a valid virtual point can be constructed. If this is not the case, the pose solution can fall back to the Aruco marker. Mitigation can consist of redundancy in the number of markers, increasing chaser capabilities (such as increasing the illumination, increasing the contrast and size of the blob, or increased camera resolution) or increasing the reliability of the blob detector in some way in future work.

A complete loss of pose estimation should only occur if the chaser navigation system fails, or if none of the found fiducial solutions produces an accurate result. If this is temporary, then the system should be able

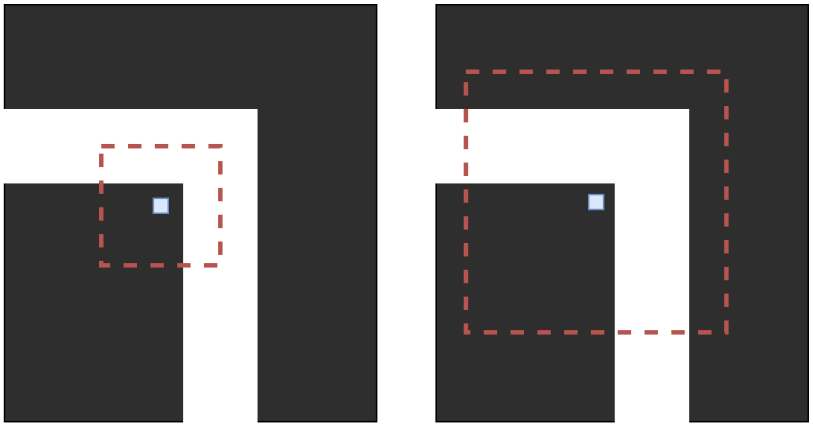


Figure 8.16: Corner ambiguity in the corner detection of the Aruco marker. The central pixel is the first estimation of the detected corner, with the red window being the search area. In the left image, no ambiguity is present since only one corner is in the search window. In the right image, the search window includes two corners and is thus ambiguous. Illustration based on [106].

to reacquire a pose when for example illumination conditions change or the relative pose is altered. While the chaser navigation system remains a single point of failure if it is not redundantly applied, with proper marker application to the target, minor marker improvements and sufficient redundancy of markers, the fiducial marker system is a robust system with mitigated risks. The effects of the proposed mitigation measures and redundant implementation of the marker system as well as the implementation of a navigation filter are shown in Figure 8.17. The decrease in severity for some failure modes is justified by the added redundancy: even though the failure mode may be present in one of the fiducials, the end-to-end system consequence is less severe if redundancy is implemented.

3			
2	5		
1	2,3	1,4	
^Likelihood Severity >	1	2	3

Figure 8.17: Risk assessment failure modes of redundant end-to-end Mantis Marker system including navigation filter and risk mitigation measures

8.4. Monte Carlo Simulation of End-to-End System Results

The Monte Carlo simulation used to derive the accuracy and precision requirements in chapter 3 was adapted to simulate the design options for the system layout proposed in section 6.1. This enables verification of the system with respect to those same requirements and the effect of the "wide" design option versus the "narrow" design option on the system performance. In addition, the end-to-end system performance is compared to a system with the same layout, but using classical planar fiducials instead of the Mantis Marker. All virtual points of the larger markers are modelled as a single protrusion in the centre of the respective markers.

Since the employed PnP solver requires at least 6 points to find a solution, it assumed that 2 virtual points of the Mantis Marker are used as feature points for the smaller central markers. This assumption is justified since it is reasonable to assume that at close range the relative attitude is small, thus bringing more virtual points in FOV.

The performance of the wide design option is shown in Figure 8.18. In Figure 8.19, the design narrow design option performance is shown. The wider design outperforms the narrow design option in terms of range performance and the number of outliers. Since the Monter Carlo model was determined to have a precision of approximately 0.1 degrees and centimetre level in section 3.7, for all rotational performance measures the difference is not significant enough to provide judgement.

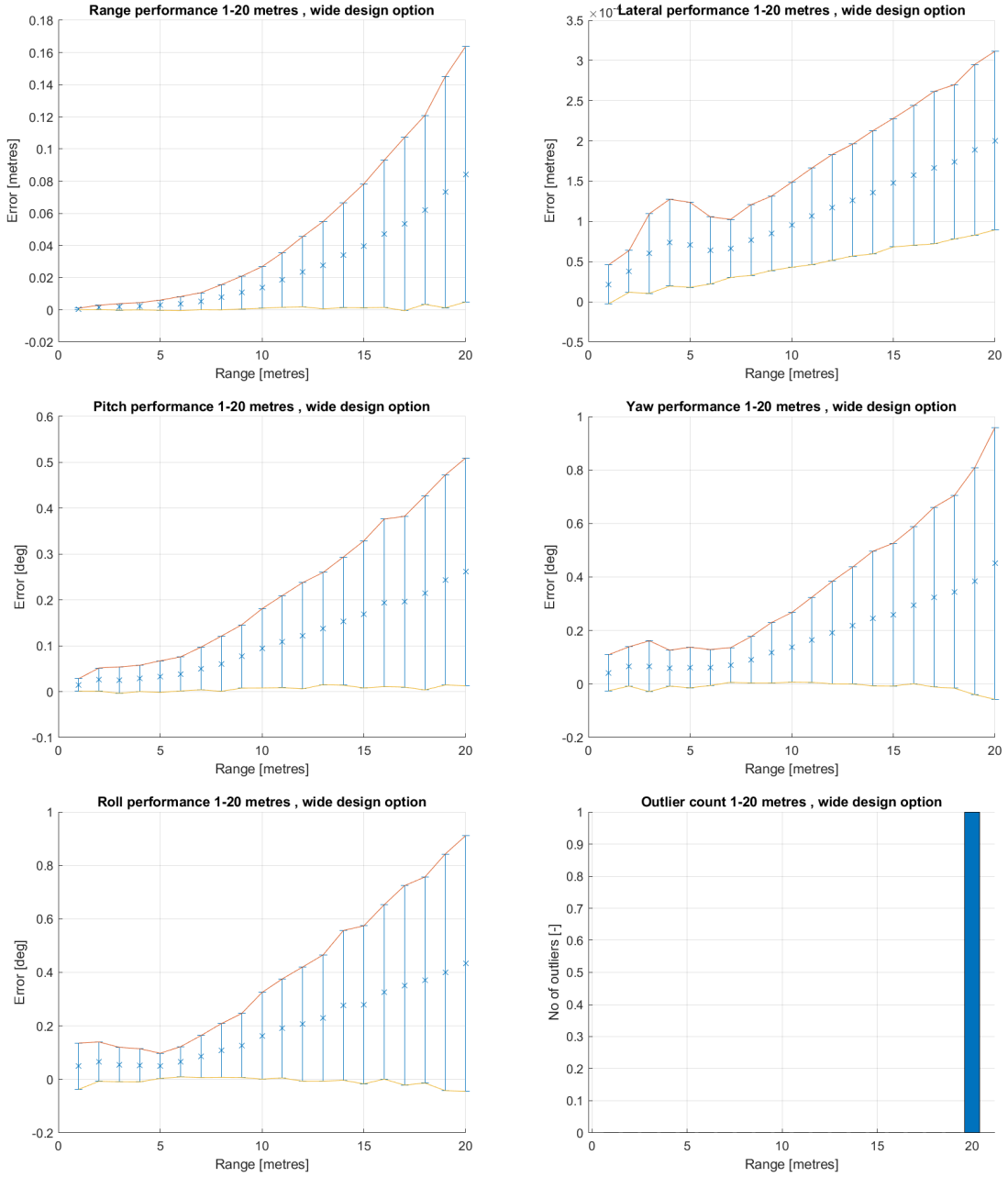


Figure 8.18: Accuracy and precision analysis for the wide design option as detailed in section 6.1. n=1500 per range step of 1 metre.

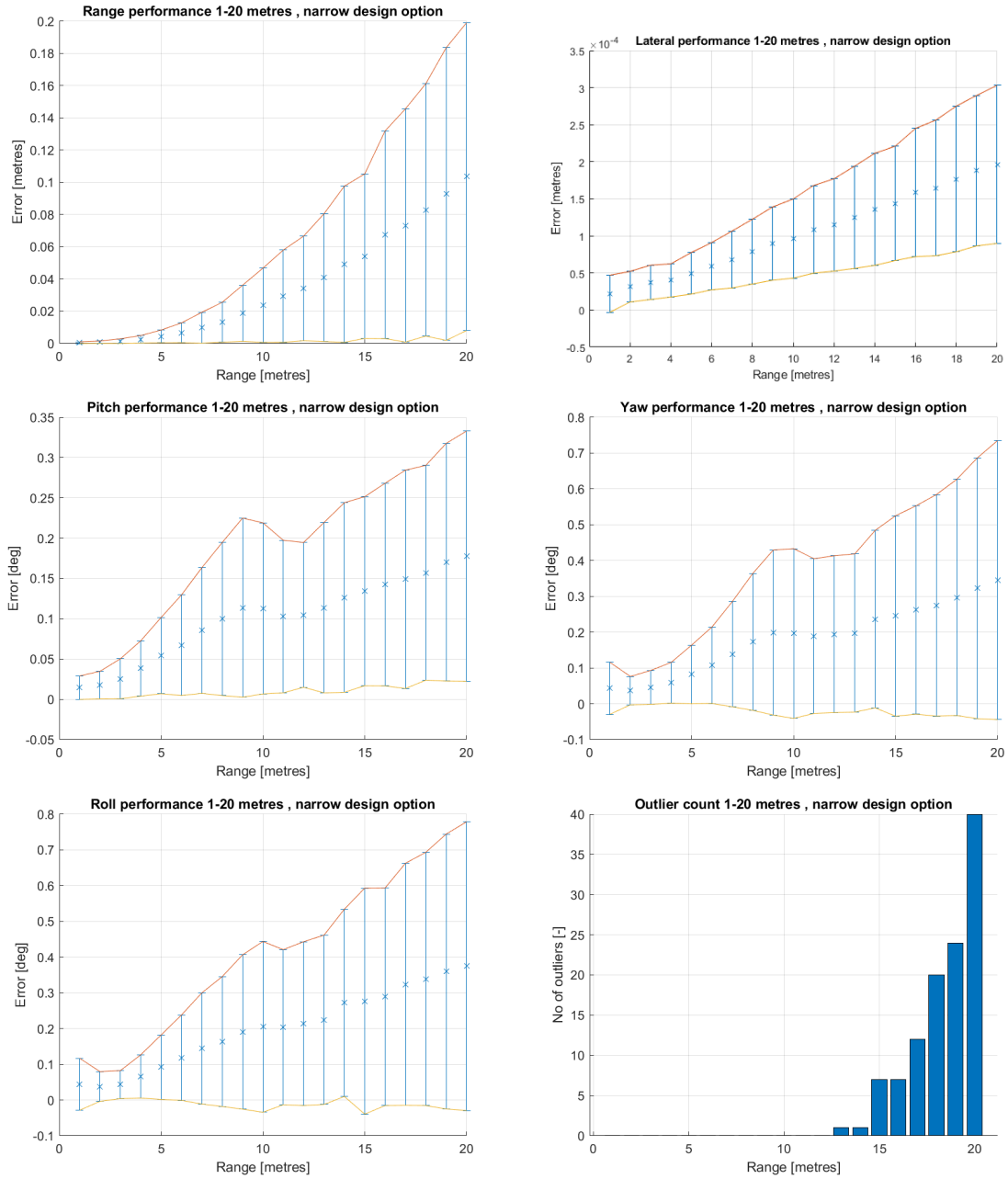


Figure 8.19: Accuracy and precision analysis for the narrow design option as detailed in section 6.1. n=1500 per range step of 1 metre.

To see what the effect of the Mantis Marker encoding element is on the performance of the end-to-end relative navigation system, the wide layout design option is modelled with all virtual points co-planar to the spacecraft face. The results are shown in Figure 8.20. Most notably, the outlier rate has increased tremendously and is constant for the entire operational envelope at 47 %. Furthermore, the range error of the solutions that are not recognised as outliers has increased significantly. Due to the large number of outliers, the accuracy of the Monte Carlo simulation is somewhat reduced, but at $n = 800$ still an accuracy of around 5 centimetres for range according to the convergence study still sufficient to judge the range performance.

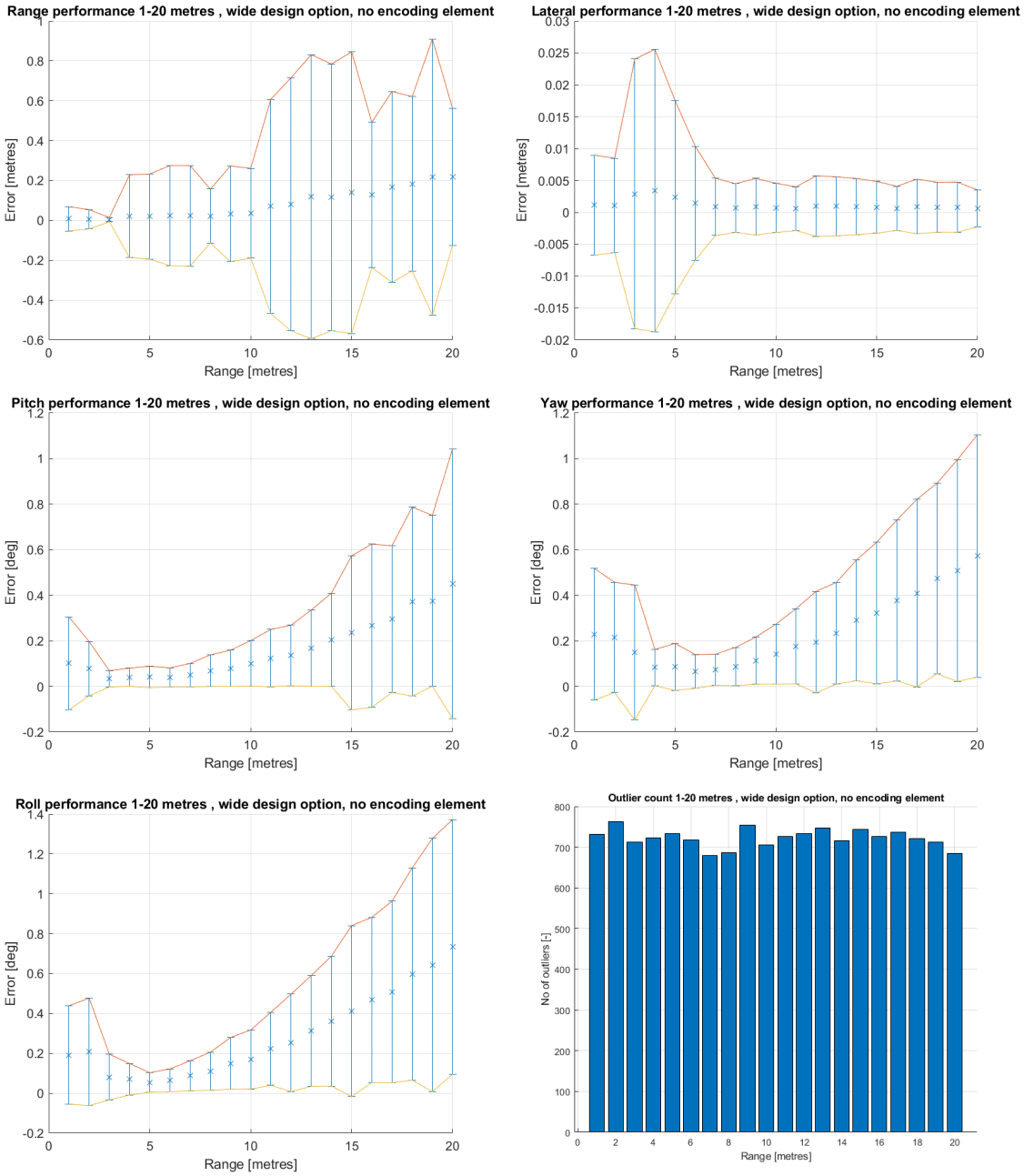


Figure 8.20: Accuracy and precision analysis for the wide design option without encoding element. n=1500 per range step of 1 metre.

8.5. System Verification and Validation

To summarise the verification of the system with respect to the system requirements, a verification matrix is presented below in Table 8.4.

System Req.	Verified	Justification
SYS-REQ-1	Yes	by analysis. Under large combined rotations the encoding element may be lost, but the planar features are still present.
SYS-REQ-2	Partially	by radiometric analysis. At long range and in eclipse, the illumination conditions are problematic.
SYS-REQ-3	Partially	by radiometric analysis. At long range and in eclipse, the illumination conditions are problematic.
SYS-REQ-4	Yes	by Monte Carlo analysis.
SYS-REQ-5	Yes	by engineering judgement. The current implementation is relatively simple and performs in real time on consumer computers.
SYS-REQ-6	Yes	by Monte Carlo analysis.
SYS-REQ-7	Yes	by Monte Carlo analysis.
SYS-REQ-8	Yes	by inspection.
SYS-REQ-9	T.B.C.	to be confirmed in future work. Currently impossible due to lack of material selection.
SYS-REQ-10	T.B.C.	to be confirmed in future work. Currently impossible due to lack of material selection.
SYS-REQ-11	Yes	by inspection, Monte Carlo analysis.
SYS-REQ-12	Yes	by engineering jugement. Due to the lack of optical parts, the Mantis Marker is straightforward to produce and qualify.
SYS-REQ-13	Yes	by experiments, Monte Carlo simulation, radiometric analysis.
SYS-REQ-14	Yes	by inspection.

Table 8.4: System verification matrix

In Table 8.5 the system validation with respect to the stakeholder requirements is presented. In future work, the system should be tested in a close loop end-to-end implementation to fully validate the Mantis Marker. However, validating the design at this stage against the stakeholder requirements shows the areas where additional work is to be done to come to a fully validated system.

Stakeh. Req.	Validated	Justification
STA-REQ-1	Yes	by experiments, Monte Carlo simulation, radiometric analysis.
STA-REQ-2	Yes	by engineering judgement.
STA-REQ-3	Yes	by inspection.
STA-REQ-4	T.B.C.	to be confirmed in future work. Currently impossible due to lack of material selection.
STA-REQ-5	T.B.C.	to be confirmed in future work. Currently impossible due to lack of material selection.
STA-REQ-6	Yes	by fulfilling SYS-REQ-11.
STA-REQ-7	Yes	by the lack of use of optical elements.
STA-REQ-8	Yes	by experiments, Monte Carlo simulation, radiometric analysis.
STA-REQ-9	Yes	by inspection.

Table 8.5: System validation matrix.

8.6. Conclusion

The results of the three performed experiments were presented in this chapter. The Blender virtual experiment showed that the Mantis Marker indeed increases the performance of the relative pose estimation with respect to planar markers. Both lack of precision in frontal observation and pose ambiguity is prevented. In terms of range, the current implementation of the Mantis Marker performs worse than the benchmark Aruco marker, but there is no reason why the corners of the Mantis Marker could not be used in the pose estimation. This would bring the range estimation up to par with the Aruco marker.

The physical experiments confirmed the behaviour of the Blender model and allowed the identification of several failure modes of the marker system. These failure modes were analysed and mitigation strategies were proposed for all. Furthermore, the physical experiment showed that the feature extraction pipeline is currently not flexible enough to function under changing illumination conditions. However, this was attributed to the tuning of both the camera and the software and is easily mitigated in the future. Both the Blender experiments as well as the physical experiments showed that the white border around the Aruco marker should be increased.

The Monte Carlo simulation results showed that the system is qualitatively able to fulfil the precision and accuracy requirements. It is expected that with the proposed improvements of the Mantis Marker, the performance of the end-to-end system will increase. The difference in theoretical performance with planar markers in the same configuration as the Mantis Marker was also tested, with the Mantis Marker outperforming the planar markers by an order of magnitude.

Finally, based on these results, the system was verified against the system requirements. Most requirements were fulfilled, although the current lack of material selection causes lifetime and space grade requirements to be needing confirmation in future work. Moreover, the illumination in eclipse is problematic at range, and some system constraints might have to be relaxed in order to meet these requirements. Validation of the system with respect to the stakeholder requirements showed that the Mantis Marker meets the customer expectations, while needing validation of the material selection in future work. In future work, the Mantis Marker should be fully validated in a closed loop end-to-end implementation.

9

Conclusion

This thesis researched encoding fiducials for space, a research field for which to the knowledge of the author no work had been published to date. The research has been driven by the central research question of this thesis: *What is the most suitable marker system for relative navigation for Active Debris Removal?*. In the process of answering the identified subquestions, a design was envisioned that required encoding elements, thus forcing the design of a novel system.

To conclude, the analyses made with respect to the research areas covered by the research questions are summarised. These questions concerned requirements (**RQ-1**), pose robustness using planar markers (**RQ-2**), system sensitivity (**RQ-3**) and system performance (**RQ-4**).

[RQ-1] Requirements for a relative navigation system for ADR

To derive appropriate requirements for a relative navigation system for ADR, a structured system engineering approach was used. A fiducial marker for ADR needs to be flexible and scalable in application in order to fulfil the system need. The system should be applicable to the LEO environment, with extreme illumination conditions (both eclipse and in sunlight). In addition, the system should be low cost, have low complexity and be completely passive.

To fulfil these requirements, it was concluded that the system should not use any optical elements nor power. The chaser elements should consist of COTS elements. The sensor should be monocular and vision-based (either visual or infrared) when costs and complexity are to be kept low. The most appropriate illumination system attempts to decrease the delta between eclipse and sunlight. This can most likely be achieved by implementing a bandpass filter on the camera together with an infrared illumination system. This system has been verified in a radiometric analysis and a physical experiment.

[RQ-2] Pose robustness using planar markers

Planar markers suffer from pose ambiguity and loss of precision in frontal observation. The former is caused by the projective similarity between two distinct poses. Under noisy conditions, this causes the pose estimation algorithm to find the wrong pose. The latter is caused by the lack of projective effects under frontal observations. This is especially true for longer distances and lenses with smaller FOV.

These effects can be mitigated by encoding additional pose information in the signal provided by the fiducial using so-called encoding markers. However, these markers are complex, not scaleable and difficult to

manufacture and qualify. To solve these issues, a simple yet effective and scaleable marker suitable for space applications is proposed: the Mantis Marker. The addition of the virtual point in the Mantis Marker was proven to increase the performance of the end-to-end system by an order of magnitude when comparing to the same system with planar markers.

[RQ-3] System sensitivity to design parameters

The system sensitivity to several design parameters was extensively analysed. The performance of the end-to-end system is dependant on a number of tradeoffs in terms of chaser sensor specifications, marker size, reflectivity, illumination, marker spacing, number of markers etc. Increased resolution of the chased sensor increases the performance of the pose estimation at higher system costs. The FOV of the chaser camera increases performance at small range but acts conversely on the performance at large range. Increasing the marker size and/or the number of markers increases the performance of the pose estimation at the cost of taking more surface space. Similarly, a wider spacing of individual markers was expected to increase the performance of the end-to-end system. However, the Monte-Carlo simulation was inconclusive in this regard.

[RQ-4] System performance of the fiducial marker system

The performance of the Mantis Marker for a single marker was benchmarked with respect to the Aruco marker. The Mantis Marker prevented the pose ambiguity and loss of precision in frontal observation present in the Aruco marker pose solutions. Furthermore, the attainable performance of the end-to-end system in a representative configuration was proven to be on par with a representative fiducial marker system for relative navigation in space. In terms of illumination, some challenges remain at range in eclipse where illumination requires a large amount of power. These challenges should be solved by relaxing certain design constraints such as illumination FOV, number of frames per second or exposure time.

Recommendations

The following recommendations are made for future work:

1. The feature extraction algorithm should be made more robust against changing illumination conditions, as explained in section 8.2.

2. The feature extraction algorithm should extract the outer corners of the fiducial to increase range performance, as explained in section 8.1.

3. The pose estimation algorithm initial guess estimation should include the virtual points to ensure faster convergence. In addition, the inefficiency caused by the unnecessary double Levenberg-Marquardt optimisation should be removed (covered in section 5.5).

4. The Mantis Marker should include a wider white border around the encoding Aruco, as explained in section 8.1.

5. Proper virtual point location calibration methods should be designed in future work, as explained in section 8.1.

6. The failure mode mitigation measures as proposed in section 8.3 should be implemented.

7. In future work, a material selection for the fiducial should be made. These materials should be qualified for the space environment. In particular, selecting and qualifying an appropriate retroreflective material is of interest. This was covered in section 6.3.

8. The design of a custom bit pattern should be investigated. This may allow a more efficient design of the marker surface. In addition, the many combinations that the Aruco marker allows for distinct identifications may not be required in ADR application. This is to be investigated more extensively in future work and was covered in section 8.1.

9. The production methods and limits for the encoding element at small scale should be investigated after material selection.

10. Appropriate application areas of the Mantis Marker in terrestrial settings are to be investigated.

Conclusion

In this thesis, a design for a novel encoding fiducial marker system was proposed. The Mantis Marker is, to the knowledge of the author of this thesis, the only encoding fiducial marker system that is able to directly encode a virtual point in the fiducial signal. This unique design was shown to increase the robustness of existing planar markers by an order of magnitude. Furthermore, it was shown that this system can be implemented in an end-to-end relative navigation system for space. The system was verified using a virtual experiment, physical experiment and a Monte Carlo simulation. The Mantis Marker is a widely applicable fiducial marker system that can facilitate ADR for future spacecraft, protecting the invaluable resource that is the access to space from Earth.

A

Code

Listing A.1: Header file of mantis marker code

```
1  //mantis_marker.h
2
3  #pragma once
4  #ifndef __MANTIS_MARKER_H_INCLUDED__
5  #define __MANTIS_MARKER_H_INCLUDED__
6
7  //Includes
8  #include <opencv2/core.hpp>
9  #include <opencv2/imgproc.hpp>
10 #include <opencv2/aruco.hpp>
11 #include <opencv2/highgui.hpp>
12 #include <opencv2/features2d.hpp>
13 #include <opencv2/calib3d.hpp>
14 #include <math.h>
15 #include <iostream>
16
17
18
19
20 namespace MantisMarker
21 {
22
23     /**
24      *@brief feature extraction of mantismarker
25      *
26      *@param input_image Image where the markers are to be found
27          *@param dictionary Aruco dictionary of the markers used in the scene
28          *@param dist_coeffs Distortion coefficients of the camera
29          *@param camera_matrix Camera intrinsics
30          *@param &points Reference to pointer of the output points
31     **/
32     void featureExtraction(cv::Mat input_image, cv::Ptr<cv::aruco::Dictionary> dictionary,
33                                             cv::Mat dist_coeffs, cv::Mat camera_matrix,
34                                             std::vector<std::vector<cv::Point2f>> &points);
35
36     /**
37      * @brief Pose estimation for single markers
38      *
39      * @param points vector of already detected markers corners and virtual mantis point on the
            image plane.
40      * For each marker, its four corners and one mantis point are provided,
41      * (e.g std::vector<std::vector<cv::Point2f> > ). For N detected markers,
42      * the dimensions of this array should be Nx5. The order of the corners should be clockwise,
            then mantispoint.
43      * @see featureExtraction
44      * @param marker_length the length of the aruco markers' side. The returning translation vectors
             will
45      * be in the same unit. Normally, unit is meters.
46      * @param mantis_depth distance between aruco marker plane and virtual mantis marker point in
            metres.
47      * @param mantis_length Longest dimension of mantis marker in metres.
48      * @param camera_matrix input 3x3 floating-point camera matrix
49      * \f$A = \vecthreethree{f_x}{0}{c_x}{0}{f_y}{c_y}{0}{0}{1}\f$
50      * @param dist_coeffs vector of distortion coefficients
```

```
51       * \f$(k_1, k_2, p_1, p_2[, k_3[, k_4, k_5, k_6],[s_1, s_2, s_3, s_4]])\f$ of 4, 5, 8 or 12
             elements
52       * @param rvecs array of output rotation vectors (@see Rodrigues) (e.g. std::vector<cv::Vec3d>).
53       * Each element in rvecs corresponds to the specific marker in imgPoints.
54       * @param tvecs array of output translation vectors (e.g. std::vector<cv::Vec3d>).
55       * Each element in tvecs corresponds to the specific marker in imgPoints.
56       *
57       * This function receives the detected markers and returns their pose estimation respect to
58       * the camera individually. So for each marker, one rotation and translation vector is returned.
59       * The returned transformation is the one that transforms points from each marker coordinate
             system
60       * to the camera coordinate system.
61       * The marker corrdinate system is centered on the middle of the marker, with the Z axis
62       * perpendicular to the marker plane.
63       * The coordinates of the four corners of the marker in its own coordinate system are:
64       * (-markerLength/2, markerLength/2, 0), (markerLength/2, markerLength/2, 0),
65       * (markerLength/2, -markerLength/2, 0), (-markerLength/2, -markerLength/2, 0)
66      **/
67      void poseEstimation(std::vector<std::vector<cv::Point2f>> points,
68                                              float marker_length, float mantis_depth, float
                                                    mantis_length,
69                                              cv::Mat camera_matrix, cv::Mat dist_coeffs,
70                                              std::vector<cv::Vec3d> &rvecs, std::vector<cv::Vec3d> &
                                                    tvecs);
71
72      /**
73       *@brief Helper function that finds the rough location of the mantis markers.
74       * Returns a vector of points, the order top left clockwise for the corners.
75       *
76       *@param corners Aruco corners
77       *@param mantis_height_factor Height of mantis expressed in aruco size
78       *@param mantis_width_factor Width of mantis expressed in aruco size
79       *@param gap_factor Gap between aruco and mantis expressed in aruco size
80      **/
81      std::vector<std::vector<cv::Point>> findMarkers(std::vector<cv::Point2f> corners,
82                                              float mantis_height_factor, float mantis_width_factor,
83                                              float gap_factor);
84  };
85
86  #endif
```

Listing A.2: Main of mantis marker code

```
1   // mantis_marker.cpp
2   #include "mantis_marker.h"
3
4   void MantisMarker::featureExtraction(cv::Mat input_image,
5                                        cv::Ptr<cv::aruco::Dictionary> dictionary,
6                                        cv::Mat dist_coeffs, cv::Mat camera_matrix,
7                                        std::vector<std::vector<cv::Point2f>> &points)
8   {
9       // Declare variables
10      cv::Mat image_gray;
11      std::vector<int> marker_ids;
12          std::vector<std::vector<cv::Point2f>> marker_corners, rejected_candidates;
13      cv::Mat distCoeffs;
14          cv::Mat cameraMatrix;
15
16          cv::Ptr<cv::aruco::DetectorParameters> parameters = cv::aruco::DetectorParameters::create();
17      //Make the detector more strict to reduce false positives
18      parameters -> errorCorrectionRate = 0.1;
19      parameters -> cornerRefinementMethod = cv::aruco::CORNER_REFINE_APRILTAG;
20
21      // Convert image to grayscale
22      cv::cvtColor(input_image, image_gray, cv::COLOR_BGR2GRAY);
23      //cv::imwrite("input.png",image_gray);
24
25
26      // Detect the aruco marker(s)
27      cv::aruco::detectMarkers(image_gray, dictionary, marker_corners,
28                                       marker_ids, parameters, rejected_candidates,
29                                       cameraMatrix = camera_matrix, distCoeffs = dist_coeffs);
30      cv::Mat outputImage = input_image.clone();
31      cv::aruco::drawDetectedMarkers(outputImage, marker_corners, marker_ids);
32      //cv::imwrite("markers.png",outputImage);
33      //cv::imshow("markers", outputImage);
34      //cv::waitKey(0);
35
36      // Check if aruco was detected
37      if (marker_corners.size()==0){
38          std::cout << "Aruco Marker not detected in Mantis feature" << std::endl;
39          return;
40      }
```

```
41      // Detect Mantis Marker based on location of aruco marker
42      for (int i = 0; i < marker_corners.size(); ++i)
43      {
44          float x_top,  x_right, y_top, y_right, x_bottom, x_left, y_bottom, y_left;
45          float x_point_1 = 0, y_point_1 = 0, x_point_2 = 0, y_point_2 = 0, x_point_3 = 0, y_point_3 =
                0, x_point_4 = 0, y_point_4 = 0;
46          bool point1 = true;
47          bool point2 = true;
48          bool point3 = true;
49          bool point4 = true;
50
51          cv::Point2f corner1 = marker_corners[i][0];
52          cv::Point2f corner2 = marker_corners[i][1];
53          cv::Point2f corner3 = marker_corners[i][2];
54          cv::Point2f corner4 = marker_corners[i][3];
55
56          double aruco_size = pow(pow((corner1.x-corner2.x),2)+pow((corner1.y-corner2.y),2),0.5);
57
58          float mantis_height_factor = 2./10.; //Mantis Marker is approx. 2/10th of aruco height
59          float mantis_width_factor = 95./100. ; // Mantis width is approx. 95/100 of aruco height
60          float gap_factor = 17./100.; //Gap is approx 17/100th of aruco height
61
62          // Mantis marker ROI corners, 4 markers, 16 corners clockwise starting top left, leave a gap
                of gap_factor
63          // Create mask for blob detection
64          std::vector<std::vector<cv::Point>> mantis_corners = MantisMarker::findMarkers(
                marker_corners[i],mantis_height_factor,mantis_width_factor,gap_factor);
65
66          for (int j = 0; j < 4; ++j)
67          {
68              cv::Mat mask = cv::Mat::zeros(input_image.rows, input_image.cols, CV_8U);
69              cv::Mat mantis_image = cv::Mat::zeros(input_image.rows, input_image.cols, CV_64FC1);
70              std::vector<std::vector<cv::Point>> mantis_corners_mask = {mantis_corners[j]};
71              cv::drawContours(mask, mantis_corners_mask, -1, cv::Scalar(128,100,100),CV_FILLED);
72              image_gray.copyTo(mantis_image,mask);
73              //cv::imwrite("MantisMask.png",mantis_image);
74              cv::Mat plot_image;
75              cv::resize(mantis_image, plot_image, cv::Size(),1,1);
76              //cv::imshow("Mantis Mask", plot_image);
77              //cv::waitKey(0);
78
79              // Blur image (convolution)
80              cv::Mat image_blurred;
81              cv::GaussianBlur(mantis_image,image_blurred,cv::Size(3,3),50,50);
82              // Now we will now find the blob
83              std::vector<cv::KeyPoint> keypoints;
84              bool not_detected = false;
85              // Set up the detector with default parameters.
86              // Setup SimpleBlobDetector parameters.
87              cv::SimpleBlobDetector::Params params;
88
89              // Change blob color to white
90              params.filterByColor = true;
91              params.blobColor = 255;
92
93              // Filter by Area
94              params.filterByArea = true;
95              params.maxArea = mantis_width_factor*mantis_height_factor*aruco_size*aruco_size*0.25;
96              params.minArea = mantis_width_factor*mantis_height_factor*aruco_size*aruco_size*0.125;
97
98              // A line has an inertia of 0, in order to prevent blob detection of the borders we
                    filter for low intertia
99              params.filterByInertia = true;
100             params.minInertiaRatio = 0.15;
101             params.maxInertiaRatio = 1;
102
103             // Make sure the mantis rasters cause no issue
104             params.minDistBetweenBlobs = round((aruco_size)/30);
105
106             params.filterByCircularity = false;
107             params.filterByConvexity = false;
108             params.thresholdStep = 1;
109             params.minThreshold = 10;
110             params.maxThreshold = 250;
111             params.minRepeatability = 2;
112
113             cv::Ptr<cv::SimpleBlobDetector> detector = cv::SimpleBlobDetector::create(params);
114             detector->detect(image_blurred, keypoints);
115
116
117             // Check if a blob was detected
118             if (keypoints.size() == 0)
119             {
120                 std::cout<< "Mantis marker not detected!" << std::endl;
```

```
121                 not_detected = true;
122             }
123             // Check if the found blob is not too close to the edge of the mantis such that it gets
                    cut off
124             if (not_detected == false)
125             {
126                 double dist_corner1 = pow(pow(keypoints[0].pt.x-mantis_corners[j][0].x,2)+pow(
                        keypoints[0].pt.y-mantis_corners[j][0].y,2),0.5);
127                 double dist_corner2 = pow(pow(keypoints[0].pt.x-mantis_corners[j][1].x,2)+pow(
                        keypoints[0].pt.y-mantis_corners[j][1].y,2),0.5);
128                 double dist_corner3 = pow(pow(keypoints[0].pt.x-mantis_corners[j][2].x,2)+pow(
                        keypoints[0].pt.y-mantis_corners[j][2].y,2),0.5);
129                 double dist_corner4 = pow(pow(keypoints[0].pt.x-mantis_corners[j][3].x,2)+pow(
                        keypoints[0].pt.y-mantis_corners[j][3].y,2),0.5);
130                 std::cout << aruco_size<< " " << dist_corner1 << std::endl;
131
132                 if(dist_corner1<aruco_size*0.175|dist_corner2<aruco_size*0.175|dist_corner3<
                        aruco_size*0.175|dist_corner4<aruco_size*0.175)
133                 {
134                     std::cout<< "Mantis marker not detected, too close to edge!" << std::endl;
135                     not_detected = true;
136                 }
137
138             }
139
140             if (j==0 && not_detected == false) // then this is the top marker
141             {
142                 x_top = keypoints[0].pt.x;
143                 y_top = keypoints[0].pt.y;
144             }
145             if (j==0 && not_detected == true)
146             {
147                 // Then we cannot calculate point 2 and 3
148                 point2 = false;
149                 point3 = false;
150             }
151             if (j==1 && not_detected == false) //right marker
152             {
153                 x_right = keypoints[0].pt.x;
154                 y_right = keypoints[0].pt.y;
155             }
156             if (j==1 && not_detected == true)
157             {
158                 // Then we cannot calculate point 1 and 2
159                 point1 = false;
160                 point2 = false;
161             }
162             if (j==2 && not_detected == false)// bottom
163             {
164                 x_bottom = keypoints[0].pt.x;
165                 y_bottom = keypoints[0].pt.y;
166             }
167             if (j==2 && not_detected == true)
168             {
169                 // Then we cannot calculate point 4 and 1
170                 point4 = false;
171                 point1 = false;
172             }
173             if (j==3 && not_detected == false) // left
174             {
175                 x_left = keypoints[0].pt.x;
176                 y_left = keypoints[0].pt.y;
177             };
178             if (j==3 && not_detected == true)
179             {
180                 // Then we cannot calculate point 3 and 4
181                 point3 = false;
182                 point4 = false;
183             }
184         }
185
186         // Calculate the slope of the sides of the aruco marker (using both sides should marginally
                increase performance)
187         float slope_h = (((corner2.y-corner1.y)+(corner3.y-corner4.y))/2.)/
188                         (((corner2.x-corner1.x)+(corner3.x-corner4.x))/2.);
189         float slope_v = (((corner4.y-corner1.y)+(corner3.y-corner2.y))/2.)/
190                         (((corner4.x-corner1.x)+(corner3.x-corner2.x))/2.);
191
192         // In the extreme case where either slope is 0 or inf, then the projection is exactly
                orthogonal
193         if(slope_h == 0  | std::isinf(slope_v))
194         {
195             if(point2 == true)
196             {
```

```
197                 x_point_2 = x_top;
198                 y_point_2 = y_right;
199             }
200 
201             if(point3 == true)
202             {
203                 x_point_3 = x_top;
204                 y_point_3 = y_left;
205             }
206 
207             if(point4 == true)
208             {
209                 x_point_4 = x_bottom;
210                 y_point_4 = y_left;
211             }
212 
213             if(point1 == true)
214             {
215                 x_point_1 = x_bottom;
216                 y_point_1 = y_right;
217             }
218 
219         }
220         else if(std::isinf(slope_h)  | slope_v == 0 )
221         {
222             if(point2 == true)
223             {
224                 x_point_2 = x_right;
225                 y_point_2 = y_top;
226             }
227 
228             if(point3 == true)
229             {
230                 x_point_3 = x_left;
231                 y_point_3 = y_top;
232             }
233 
234             if(point4 == true)
235             {
236                 x_point_4 = x_left;
237                 y_point_4 = y_bottom;
238             }
239 
240             if(point1 == true)
241             {
242                 x_point_1 = x_right;
243                 y_point_1 = y_bottom;
244             }
245 
246         }
247         else
248         {
249             // Calculate the x of the virtual point of the mantis marker in the image plane
250             // Calculate b for y_top = slope_v*x_top+b
251             float b1 = y_top-(slope_v*x_top);
252             float b2 = y_right-(slope_h*x_right);
253             if(point2 == true)
254             {
255                 x_point_2 = (b1-b2)/(slope_h-slope_v);
256                 y_point_2 = slope_v*x_point_2+b1;
257             }
258 
259             b2 = y_left-(slope_h*x_left);
260 
261             if(point3 == true)
262             {
263                 x_point_3 = (b1-b2)/(slope_h-slope_v);
264                 y_point_3 = slope_v*x_point_3+b1;
265             }
266 
267             b1 = y_bottom-(slope_v*x_bottom);
268 
269             if(point4 == true)
270             {
271                 x_point_4 = (b1-b2)/(slope_h-slope_v);
272                 y_point_4 = slope_v*x_point_4+b1;
273             }
274 
275             b2 = y_right-(slope_h*x_right);
276 
277             if(point1 == true)
278             {
279                 x_point_1 = (b1-b2)/(slope_h-slope_v);
280                 y_point_1 = slope_v*x_point_1+b1;
```

```
281             }
282         }
283         cv::Point2f mantis_point_1(x_point_1,y_point_1);
284         cv::Point2f mantis_point_2(x_point_2,y_point_2);
285         cv::Point2f mantis_point_3(x_point_3,y_point_3);
286         cv::Point2f mantis_point_4(x_point_4,y_point_4);
287 
288         points.push_back(std::vector<cv::Point2f>({corner1,corner2,corner3,corner4,
289                     mantis_point_1,mantis_point_2,mantis_point_3,mantis_point_4}));
290 
291         // Plot the point
292         cv::Point centerCircle1(x_top,y_top);
293         cv::Point centerCircle2(x_right,y_right);
294         cv::Point centerCircle3(x_bottom,y_bottom);
295         cv::Point centerCircle4(x_left,y_left);
296 
297         int radiusCircle = 2;
298         cv::Scalar colorCircle2(0,100,0);
299         cv::Scalar colorCircle3(255,0,0);
300         cv::Scalar colorCircle4(0,0,100);
301         cv::Scalar colorCircle5(255,255,0);
302         cv::circle(input_image, mantis_point_1, radiusCircle, colorCircle2, CV_FILLED);
303         cv::circle(input_image, mantis_point_2, radiusCircle, colorCircle2, CV_FILLED);
304         cv::circle(input_image, mantis_point_3, radiusCircle, colorCircle2, CV_FILLED);
305         cv::circle(input_image, mantis_point_4, radiusCircle, colorCircle2, CV_FILLED);
306         cv::circle(input_image, centerCircle1, radiusCircle, colorCircle2, CV_FILLED);
307         cv::circle(input_image, centerCircle2, radiusCircle, colorCircle2, CV_FILLED);
308         cv::circle(input_image, centerCircle3, radiusCircle, colorCircle2, CV_FILLED);
309         cv::circle(input_image, centerCircle4, radiusCircle, colorCircle2, CV_FILLED);
310         cv::circle(input_image, corner1, radiusCircle, colorCircle2, CV_FILLED);
311         cv::circle(input_image, corner2, radiusCircle, colorCircle3, CV_FILLED);
312         cv::circle(input_image, corner3, radiusCircle, colorCircle4, CV_FILLED);
313         cv::circle(input_image, corner4, radiusCircle, colorCircle5, CV_FILLED);
314 
315         //cv::imwrite("points.png",input_image);
316         //cv::imshow("point", input_image);
317         //cv::waitKey(0);
318 
319     };
320 
321 };
322 
323 void MantisMarker::poseEstimation(std::vector<std::vector<cv::Point2f>> points,
324                                   float marker_length, float mantis_depth, float
                                        mantis_length,
325                                   cv::Mat camera_matrix, cv::Mat dist_coeffs,
326                                   std::vector<cv::Vec3d> &rvecs, std::vector<cv::Vec3d> &
                                        tvecs)
327     {
328         CV_Assert(marker_length > 0 & mantis_depth > 0);
329 
330         // Create initialising array for cvMat of initial solution in cvFindExtrinsicCameraParams2
331         float obj_point_ar[3][4] = {{-marker_length / 2.f, marker_length / 2.f, marker_length / 2.f,
                -marker_length / 2.f},
332                                     {marker_length / 2.f, marker_length / 2.f, -marker_length / 2.f,
                                          -marker_length / 2.f},
333                                     {0,0,0,0}};
334 
335         int n_markers = points.size();
336         rvecs = std::vector<cv::Vec3d>(n_markers);
337         tvecs = std::vector<cv::Vec3d>(n_markers);
338 
339 
340         // for each marker, calculate its pose
341         cv::parallel_for_(cv::Range(0, n_markers), [&](const cv::Range& range) {
342             const int begin = range.start;
343             const int end = range.end;
344 
345             for (int i = begin; i < end; ++i)
346             {
347                 std::vector<cv::Vec3f> obj_points;
348                 // Get object points of aruco
349                 // set coordinate system in the middle of the marker, with Z pointing out
350                 obj_points.push_back(cv::Vec3f(-marker_length / 2.f, marker_length / 2.f, 0));
351                 obj_points.push_back(cv::Vec3f(marker_length / 2.f, marker_length / 2.f, 0));
352                 obj_points.push_back(cv::Vec3f(marker_length / 2.f, -marker_length / 2.f, 0));
353                 obj_points.push_back(cv::Vec3f(-marker_length / 2.f, -marker_length / 2.f, 0));
354 
355                 cv::Mat rvec, tvec;
356                 rvec.create(3, 1, CV_64FC1);
357                 tvec.create(3, 1, CV_64FC1);
358                 cv::Mat ini_obj(3,4,CV_32F,obj_point_ar);
359                 float img_points_ar[2][4] = {{points[i][0].x,points[i][1].x,points[i][2].x,points[i
                        ][3].x},
```

```
360                                         {points[i][0].y,points[i][1].y,points[i][2].y,points[i
                                                ][3].y}};
361                 cv::Mat img_points(2,4,CV_32F,img_points_ar);
362
363
364                 //Provide initial guess of pose using the aruco marker
365                 //Function is somewhat outdated and needs deprecated types
366                 CvMat c_objectPoints = cvMat(ini_obj), c_imagePoints = cvMat(img_points);
367                 CvMat c_cameraMatrix = cvMat(camera_matrix), c_distCoeffs = cvMat(dist_coeffs);
368                 CvMat c_rvec = cvMat(rvec), c_tvec = cvMat(tvec);
369                 cvFindExtrinsicCameraParams2(&c_objectPoints, &c_imagePoints, &c_cameraMatrix,
370                                         (c_distCoeffs.rows && c_distCoeffs.cols) ? &c_distCoeffs
                                                : 0,
371                                         &c_rvec, &c_tvec);
372                 rvecs[i] = rvec;
373                 tvecs[i] = tvec;
374                 // Create a copy of the points vector in order to dynamically populate the
                        pnp_points vector and remove zero entries
375                 std::vector<cv::Point2f>  pnp_points;
376                 // Make sure only one point is added in order to prevent overfitting for range
377                 bool point_added = false;
378                 for(int j = 0; j < 4; ++j){pnp_points.push_back(points[i][j]);};
379                 // Check which virtual points were found
380                 if(points[i][4].x != 0){
381                     // add point 1 to the obj_points vector & pnp_vector
382                     obj_points.push_back(cv::Vec3f(-0.25*mantis_length,0.25*mantis_length, -
                            mantis_depth));
383                     pnp_points.push_back(points[i][4]);
384                     point_added = true;
385                 }
386                 if(points[i][5].x != 0 && point_added == false){
387                     // add point 2
388                     obj_points.push_back(cv::Vec3f(0.25*mantis_length,0.25*mantis_length, -
                            mantis_depth));
389                     pnp_points.push_back(points[i][5]);
390                     point_added = true;
391                 }
392                 if(points[i][6].x != 0 && point_added == false){
393                     // add point 3
394                     obj_points.push_back(cv::Vec3f(0.25*mantis_length,-0.25*mantis_length, -
                            mantis_depth));
395                     pnp_points.push_back(points[i][6]);
396                     point_added = true;
397                 }
398                 if(points[i][7].x != 0 && point_added == false){
399                     // add point 4
400                     obj_points.push_back(cv::Vec3f(-0.25*mantis_length,-0.25*mantis_length, -
                            mantis_depth));
401                     pnp_points.push_back(points[i][7]);
402                     point_added = true;
403                 }
404
405                 // Solve PnP
406                 for(int i = 0; i< obj_points.size(); ++i){
407                     std::cout << obj_points[i] <<std::endl;
408                 }
409                 std::cout << pnp_points << std::endl;
410                 cv::solvePnP(obj_points, pnp_points, camera_matrix, dist_coeffs, rvecs[i],
411                             tvecs[i],true);
412
413                 //std::cout <<  rvecs[i] << " " << obj_points.size()<< " " << points[i].size()<< std
                        ::endl;
414             };
415         });
416     };
417
418 std::vector<std::vector<cv::Point>> MantisMarker::findMarkers(std::vector<cv::Point2f> corners,
        float mantis_height_factor, float mantis_width_factor,
419     float gap_factor)
420         {
421
422                 // 4 points per marker (vector is Nx4)
423                 cv::Point2f corner1,corner2,corner3,corner4;
424
425                 std::vector<std::vector<cv::Point>> mantis_corners_vector;
426
427                 cv::Point mantis_corner;
428
429                 for (int i = 0; i < 4; i++)
430                 {
431                         std::vector<cv::Point> mantis_corners;
432                         double margin = 0.05; // Five percent margin on each side
433
434                         if (i==0){corner1 = corners[0]; corner2 = corners[1]; corner3 = corners[2];
```

```
                                corner4 = corners[3];};
435                             if (i==1){corner1 = corners[1]; corner2 = corners[2]; corner3 = corners[3];
                                    corner4 = corners[0];};
436                             if (i==2){corner1 = corners[2]; corner2 = corners[3]; corner3 = corners[0];
                                    corner4 = corners[1];};
437                             if (i==3){corner1 = corners[3]; corner2 = corners[0]; corner3 = corners[1];
                                    corner4 = corners[2];};
438
439                             mantis_corner.x = corner1.x+(corner1.x-corner4.x)*(mantis_height_factor +
                                    gap_factor - margin);
440                             mantis_corner.y = corner1.y+(corner1.y-corner4.y)*(mantis_height_factor +
                                    gap_factor - margin);
441                             mantis_corners.push_back(mantis_corner);
442
443                             mantis_corner.x = corner2.x+(corner2.x-corner3.x)*(mantis_height_factor +
                                    gap_factor - margin);
444                             mantis_corner.y = corner2.y+(corner2.y-corner3.y)*(mantis_height_factor +
                                    gap_factor - margin);
445                             mantis_corners.push_back(mantis_corner);
446
447                             mantis_corner.x = corner2.x+(corner2.x-corner3.x)*(gap_factor + margin);
448                             mantis_corner.y = corner2.y+(corner2.y-corner3.y)*(gap_factor + margin);
449                             mantis_corners.push_back(mantis_corner);
450
451                             mantis_corner.x = corner1.x+(corner1.x-corner4.x)*(gap_factor + margin);
452                             mantis_corner.y = corner1.y+(corner1.y-corner4.y)*(gap_factor + margin);
453                             mantis_corners.push_back(mantis_corner);
454
455                             mantis_corners_vector.push_back(mantis_corners);
456                 };
457
458                 return(mantis_corners_vector);
459
460     };
```

Listing A.3: Camera calibration script for C++ pipeline

```
1
2   #include <opencv2/opencv.hpp>
3   #include <opencv2/calib3d/calib3d.hpp>
4   #include <opencv2/highgui/highgui.hpp>
5   #include <opencv2/imgproc/imgproc.hpp>
6   #include <stdio.h>
7   #include <iostream>
8
9   #include "opencv2/highgui.hpp"
10  #include "opencv2/calib3d.hpp"
11
12  // Defining the dimensions of checkerboard
13  const int CHECKERBOARD[2]{ 9,6 };
14  const float squareSize = 0.075;
15
16  int main()
17  {
18      // Creating vector to store vectors of 3D points for each checkerboard image
19      std::vector<std::vector<cv::Point3f> > objpoints;
20
21      // Creating vector to store vectors of 2D points for each checkerboard image
22      std::vector<std::vector<cv::Point2f> > imgpoints;
23
24      // Defining the world coordinates for 3D points
25      std::vector<cv::Point3f> objp;
26
27      for (int i{ 0 }; i < CHECKERBOARD[1]; i++)
28      {
29          for (int j{ 0 }; j < CHECKERBOARD[0]; j++)
30              objp.push_back(cv::Point3f(j*squareSize, i*squareSize, 0));
31
32      }
33
34
35      // Extracting path of individual image stored in a given directory
36      std::vector<cv::String> images;
37      // Path of the folder containing checkerboard images
38      std::string path = "/data/Datasets/run8/";
39
40      cv::glob(path, images);
41
42      cv::Mat frame, gray, gray_small;
43      // vector to store the pixel coordinates of detected checker board corners
44      std::vector<cv::Point2f> corner_pts;
45      bool success;
46
```

```
47      // Looping over all the images in the directory
48      for (int i{ 0 }; i < images.size(); i++)
49      {
50          std::cout << images[i] << std::endl;
51          std::cout << "Starting Calibration" << std::endl;
52          frame = cv::imread(images[i]);
53          cv::cvtColor(frame, gray, cv::COLOR_BGR2GRAY);
54
55          //Downsize image to find initial corner coordinates
56          cv::resize(gray, gray_small, cv::Size(), 0.2, 0.2);
57
58          // Finding checker board corners
59          // If desired number of corners are found in the image then success = true
60          success = cv::findChessboardCorners(gray_small, cv::Size(CHECKERBOARD[0], CHECKERBOARD[1]),
                corner_pts, cv::CALIB_CB_ADAPTIVE_THRESH);
61          std::cout << "Finished finding corners" << std::endl;
62          /*
63           * If desired number of corner are detected,
64           * we refine the pixel coordinates and display
65           * them on the images of checker board
66          */
67          if (success)
68          {
69              std::cout << "Success" << std::endl;
70              cv::TermCriteria criteria(cv::TermCriteria::EPS | cv::TermCriteria::MAX_ITER, 30, 0.001)
                    ;
71              // Scale back the corner points to original resolution
72              for (int i = 0;i<corner_pts.size();++i){corner_pts[i].x = corner_pts[i].x *5; corner_pts
                    [i].y = corner_pts[i].y*5;};
73              // refining pixel coordinates for given 2d points.
74              // resize calibration image
75              cv::resize(gray, gray_small, cv::Size(), 1, 1);
76              cv::cornerSubPix(gray, corner_pts, cv::Size(11, 11), cv::Size(-1, -1), criteria);
77
78              // Displaying the detected corner points on the checker board
79              //cv::drawChessboardCorners(frame, cv::Size(CHECKERBOARD[0], CHECKERBOARD[1]),
                    corner_pts, success);
80
81              objpoints.push_back(objp);
82              imgpoints.push_back(corner_pts);
83          }
84          else
85          {
86              std::cout << "Failed" << std::endl;
87          }
88          //cv::imshow("Image", frame);
89          //cv::waitKey(0);
90      }
91
92      cv::destroyAllWindows();
93
94      cv::Mat cameraMatrix, distCoeffs;
95      std::vector<cv::Mat> R, T;
96
97      /*
98       * Performing camera calibration by
99       * passing the value of known 3D points (objpoints)
100      * and corresponding pixel coordinates of the
101      * detected corners (imgpoints)
102     */
103     cv::calibrateCamera(objpoints, imgpoints, cv::Size(gray.rows, gray.cols), cameraMatrix,
            distCoeffs, R, T);
104
105     std::cout << "cameraMatrix : " << cameraMatrix << std::endl;
106     std::cout << "distCoeffs : " << distCoeffs << std::endl;
107
108     // Print translation for quick check of solution
109     for (std::vector<cv::Mat>::const_iterator i = T.begin(); i != T.end(); ++i)
110         std::cout << *i << ' ';
111
112     // Compute reprojection error
113
114     std::vector<cv::Point2f> imagePoints2;
115     size_t totalPoints = 0;
116     double totalErr = 0, err;
117
118     for (size_t i = 0; i < objpoints.size(); ++i)
119     {
120
121         projectPoints(objpoints[i], R[i], T[i], cameraMatrix, distCoeffs, imagePoints2);
122
123         err = norm(imgpoints[i], imagePoints2, cv::NORM_L2);
124         size_t n = objpoints[i].size();
125         totalErr += err * err;
```

```
126             totalPoints += n;
127         }
128         float meanError = std::sqrt(totalErr / totalPoints);
129         std::cout << "Mean reprojection error in pixels : " << meanError << std::endl;
130 
131         if (meanError<1)
132         {
133             // Save calibration result to file
134             std::cout << "Writing camera parameters to file..." << std::endl;
135 
136             cv::FileStorage fs("cam_params_long_6000_60_percent.xml", cv::FileStorage::WRITE);
137             fs << "cameraMatrix" << cameraMatrix;
138             fs << "distCoeffs" << distCoeffs;
139             fs.release();
140         }
141         else
142         {
143             std::cout << "Calibration failed: reprojection error above defined threshold!" << std::endl;
144         }
145 
146         return 0;
147     }
```

Listing A.4: Matlab code Monte Carlo Simulation

```
1   % Monte Carlo simulation for end-to-end system model RENDER - David Rijlaarsdam
2   % davidrijlaarsdam@outlook.com
3   
4   % Disable warning that slows excecution
5   warning('off','MATLAB:nearlySingularMatrix');
6   
7   % Load camera parameters
8   F = 20.19; %mm
9   Sx = 1/(8.6*10^(-3));
10  Sy = 1/(8.3*10^(-3));
11  Px = 752;
12  Py = 580;
13  
14  % Set titles
15  run_title = "wide_option_flat";
16  design_option = ", wide design option, no encoding element";
17  
18  
19  IntrinsicMatrix = [F*Sx 0 0; 0 F*Sy 0; Px/2. Py/2. 1];
20  imageSize = [Px,Py];
21  radialDistortion = [0 0];
22  cameraParams = cameraParameters('IntrinsicMatrix',IntrinsicMatrix,'RadialDistortion',radialDistortion, 'ImageSize',
        imageSize);
23  % Take the tranpsose in order to get intrinsic matrix in standard notation
24  intrinsicMatrix = cameraParams.IntrinsicMatrix';
25  focalLength = transpose(cameraParams.FocalLength); % In pixels
26  
27  centerMarkerWidth = 0.025; % m. Half of 5 cm since aruco is smaller than entire marker
28  centerMarkerDepth = 0;
29  centerMarkerLocations = [-0.075,-0.075,0;0,0.075,0;0.075,-0.075,0;0.1,0.1,0];
30  
31  markerWidth = 0.05; % m. Half of 10 cm since aruco is smaller than entire marker
32  markerDepth = 0;
33  markerLocations = [-0.4,0.4,0;0.4,0.4,0;0.4,-0.4,0;-0.4,-0.4,0];
34  
35  noPoints = 8; % Number of points
36  maxSpacing = 0.5; % Maximum spacing between outer points in metres
37  minSpacing = 0.01; % Minimum spacing between points in metres
38  maxVertical = 0.1; % Maximum vertical spacing for points
39  
40  simNumber = 1500;
41  
42  posNoise = [0.1,0.5]; % Positional noise std in pixels for [1, 100] metres
43  minRange = 1; % Minimum range of simulation, metres
44  maxRange = 20; % Maximum range of simulation, metres
45  rangeStep = 1; % Stepsize for range calculation, metres
46  rotRange = [-45,45]; % Rotational range in pitch, yaw ([min,max]). Deg.
47  
48  % Create all combinations and prepopulate the scene variables
49  range = minRange:rangeStep:maxRange;
50  rotation1 = unifrnd(rotRange(1),rotRange(2),simNumber*length(range),1);
51  rotation2 = unifrnd(rotRange(1),rotRange(2),simNumber*length(range),1);
52  roll = unifrnd(-180,180,simNumber*length(range),1);
53  scenes = [repelem(range,simNumber)',rotation1,rotation2,roll];
54  
55  % Ask if we should proceed
56  approve = input(['Number of simulations is: ',num2str(length(scenes)),'. Proceed? Y/N [Y]:'],'s');
57  if approve == 'N'
```

```
    disp('Simulation cancelled');
    return
end
disp('Starting simulation...');

resultScenes = zeros(simNumber,length(scenes(1,:)));
resultR = zeros(simNumber*3,3);
resultt = zeros(simNumber*3,1);

rotationPlotEpnp = zeros(length(scenes(:,1)*simNumber),3);
z = 1;
% Start model loop - Iterate over all scenes
for i = 1:length(scenes(:,1))
   disp(i);
   R = eul2rotm([deg2rad(scenes(i,2)),deg2rad(scenes(i,3)),deg2rad(scenes(i,4))]);
   t = [0,0,scenes(i,1)];
   j = false;
   while j == false
       % Calculate world coordinates of points
       % points = generatePoints(noPoints,maxSpacing,minSpacing,maxVertical);
       % Calculate world coordinates of points
       points = [];
       for k = 1:length(markerLocations(:,1))
            markerPoints = [-markerWidth/2,markerWidth/2,0;markerWidth/2,markerWidth/2,0;
                markerWidth/2,-markerWidth/2,0;-markerWidth/2,-markerWidth/2,0;0,0,markerDepth];
            points = [points;repmat(markerLocations(k,:),5,1)+markerPoints];
       end

        % Add one small marker for close range with 2 virtual points added

       for k = 1:length(centerMarkerLocations(:,1))
            centerMarkerPoints = [-centerMarkerWidth/4,centerMarkerWidth/4,0;centerMarkerWidth/4,centerMarkerWidth/4,0;
                centerMarkerWidth/4,-centerMarkerWidth/4,0;-centerMarkerWidth/4,-centerMarkerWidth/4,0;
                -centerMarkerWidth/8,0,centerMarkerDepth;+centerMarkerWidth/8,0,centerMarkerDepth];
            points = [points;repmat(centerMarkerLocations(k,:),6,1)+centerMarkerPoints];

       end


       % Calculate projections of points
       projectedPoints = worldToImage(cameraParams,R,t,points);
       % Make sure to only include points that are within FOV
       points = points(0<projectedPoints(:,1) & projectedPoints(:,1)<cameraParams.ImageSize(2),:);
       projectedPoints = projectedPoints(0<projectedPoints(:,1) & projectedPoints(:,1)<cameraParams.ImageSize(2),:);
       points = points(0<projectedPoints(:,2) & projectedPoints(:,2)<cameraParams.ImageSize(1),:);
       projectedPoints = projectedPoints(0<projectedPoints(:,2) & projectedPoints(:,2)<cameraParams.ImageSize(1),:);

       % Calculate noise level at this range
       % 'max centroiding error is calculated according to Calhoun et al. (1995):
       % Solution to the problem of determining the relative 6 DOF state for spacecraft
       % automated rendezvous and docking'
       noise = posNoise(1)+(posNoise(2)-posNoise(1))*(scenes(i,1)/100)^3;
       % Add Guassian positional noise
       projectedPoints = projectedPoints+ noise*randn(size(projectedPoints)); %This noise level is in x and y

%        figure;
%        plot(projectedPoints(:,1),projectedPoints(:,2),"x");
%        xlim([0 752]);
%        ylim([0 580]);

       % Perform PnP(s)
       %=======================EPNP with Gauss-Newton====================
       % Check if there are at least 6 points in the frame
       if length(projectedPoints(:,1))<6
           disp('Not enough points for EPNP');
           epnpR = NaN(3,3);
           epnpt = NaN(3,1);
           % pause;
       else
           % Transform into homogenous coordinates
           markers = [projectedPoints,ones(length(projectedPoints),1)];
           markersWorldHomo = [points,ones(length(points),1)];
           try
               [epnpR,epnpt,Xc,best_solution,~] = efficient_pnp_gauss(markersWorldHomo,markers,intrinsicMatrix);
              % Generate indices for result matrices
               resultLine = (i-1)*3+1;

               resultScenes(i,:) = scenes(i,:); % Duplicate so we can adapt this vector without losing scene info
               resultR(resultLine:resultLine+2,:) = [epnpR]; %,p3pR,mlpnpR];
               resultt(resultLine:resultLine+2,:) = [epnpt]; %,p3pt,mlpnpt];
               rotationPlotEpnp(z,:) = rad2deg(rotm2eul(epnpR, "XYZ"));
               j = true;
               z = z+1;
           catch
```

```
                disp('Error in epnp, perhaps not enough inliers or not in path?')
                epnpR = NaN(3,3);
                epnpt = NaN(3,1);
            end
        end
    end
end

disp('Finished');

% Calculate error and number of outliers in the error
outliers = zeros(1,length(range));

% repelem of the simNumber since we calculate every scene simNumber
% times
error_pitch = abs(scenes(:,2)+rotationPlotEpnp(:,3));
error_pitch(error_pitch>170) = 180-error_pitch(error_pitch>170);
error_yaw = abs(scenes(:,3)+rotationPlotEpnp(:,2));
error_yaw(error_yaw>170) = 180-error_yaw(error_yaw>170);
error_roll = abs(scenes(:,4)+rotationPlotEpnp(:,1));
error_roll(error_roll>350) = 360-error_roll(error_roll>350);

error_lateral = sqrt(resultt(1:3:end,1).^2+resultt(2:3:end,1).^2);
error_range = abs(resultScenes(:,1)-resultt(3:3:end,1));

% If the range error is above the full range due to flipped solution,
% we consider this solution an outlier. Likewise, if the roll error is
% extremely large, we discard the solution and count the outlier.
i = 1;
while i<=length(error_range)
    if error_range(i)>resultScenes(i,1) || error_roll(i)>60
        error_range(i) = [];
        error_pitch(i) = [];
        error_yaw(i) = [];
        error_roll(i) = [];
        error_lateral(i) = [];
        outliers(range==resultScenes(i,1)) = outliers(range==resultScenes(i,1))+1;
        resultScenes(i,:) = [];
    else
        i = i+1;
    end
end

% Calculate mean and std
std_pitch = zeros(length(range),1);
std_yaw = zeros(length(range),1);
std_roll = zeros(length(range),1);
std_lateral = zeros(length(range),1);
std_range = zeros(length(range),1);

mean_pitch = zeros(length(range),1);
mean_yaw = zeros(length(range),1);
mean_roll = zeros(length(range),1);
mean_lateral = zeros(length(range),1);
mean_range = zeros(length(range),1);

j = 1;

for i=minRange:maxRange
    std_pitch(j) = std(error_pitch(resultScenes(:,1)==i),'omitnan');
    std_yaw(j) = std(error_yaw(resultScenes(:,1)==i),'omitnan');
    std_roll(j) = std(error_roll(resultScenes(:,1)==i),'omitnan');
    std_lateral(j) = std(error_lateral(resultScenes(:,1)==i),'omitnan');
    std_range(j) = std(error_range(resultScenes(:,1)==i),'omitnan');

    mean_pitch(j) = mean(error_pitch(resultScenes(:,1)==i),'omitnan');
    mean_yaw(j) = mean(error_yaw(resultScenes(:,1)==i),'omitnan');
    mean_roll(j) = mean(error_roll(resultScenes(:,1)==i),'omitnan');
    mean_lateral(j) = mean(error_lateral(resultScenes(:,1)==i),'omitnan');
    mean_range(j) = mean(error_range(resultScenes(:,1)==i),'omitnan');

    j=j+1;
end
```

Listing A.5: Matlab Code Radiometric Analysis

```
% Radiometric Analysis – 2D simplified model of optical properties of relative
% navigation system

range = 20; % metres
sizeMarker = 0.05;
areaMarker = sizeMarker^2; % m^2, area of individual marker
sizeSpacecraft = 1;
```

```
areaSpacecraft = sizeSpacecraft^2; %metre
numberLEDS = 100; % Number of leds used
detectorSize = 752;
pixelSize = 8.6*10^(-3);
sizeBit = sizeMarker/10;
areaBit = (sizeBit)^2; % m^2

theta = 0; % Angle of incidence of incoming light w.r.t. marker plane, rad
lightCone = 8; % Angle of lightcone, full
fov = 16; % FOV camera, deg
focalLength = 20.19*10^-3; % Focal length camera, m
fStop = 2.8; % F# camera
diafragm = focalLength/fStop;
exposureTime = 0.001; % Exposure time in seconds
throughput = 0.35; % The relative percentage of the total light that reaches the detector after interacting with all the optical elements of the instrument, otherwise known as the optical throughput.
Qe = 0.35; % Quantum efficiency detector, e/photon (according to prosilica datasheet)
lambda = 850*10^-9; % Wavelenght of light from lamp
c = physconst('LightSpeed');
planck = 6.62607015*10^-34;% J*s

powerUsed = 2.1*numberLEDS*30*exposureTime; % Power of leds for SFH 4783


fov = deg2rad(fov);
lightCone = deg2rad(lightCone);

upperBand = 900;
lowerBand = 800;
spectralIrradianceSun = 0;
irradianceSun = (upperBand-lowerBand)*spectralIrradianceSun; % W/m^2

reflectivitySpacecraft = 10; % sr^-1 Reflectance Measurements of Multi-Layer Insulation (MLI) (degrades according to Modelling of Solar Radiation Pressure Effects: Parameter Analysis for the MICROSCOPE Mission)
reflectivityMarker = 200; % sr^-1 for Conspicuity tape for enhanced laser range finding Show sensitivity

snratio = zeros(20,4);
powerRatio = zeros(20,1);
signalSpacecraft = zeros(20,1);
signalBit = zeros(20,1);
bitPixels = zeros(20,1);
scPixels = zeros(20,1);
S = zeros(20,1);
fullS = zeros(20,1);
shotNoise = zeros(20,1);
totalNoise = zeros(20,1);

for i = 1:20
    z = 1;
    for j = 50:50:200
        exposureTime = 1/j;
        bitPixels(i) = atan(sizeBit/2/i)/(fov/2)*detectorSize;
        scPixels(i) =   atan(sizeSpacecraft/2/i)/(fov/2)*detectorSize;
        %     if bitPixels(i)<1
        %         bitPixels(i) = 1;
        %     end
        srCamera = diafragm^2*pi/(i^2);

        %==============Marker Light=========================
        r = atan(lightCone/2)*i; % Assume no spherical cap
        Slight = pi*r^2;
        srLight = Slight/i^2; % Steradians of light cone (constant)
        r_catalog = atan(24/2)*i;
        Slight_catalog = pi*r_catalog^2;
        srLightCatalog = Slight_catalog/i^2;
        phi = 3.5*numberLEDS; % Radiant flux emmited by light source W/sr, modelled as point source SFH 4783
        phi = phi*srLightCatalog/srLight; % times catalog sterradians divided by new sterradians
        omega = areaBit/(i^2); % Solid angle of bit from lightsource, steradians
        omegaSc = areaSpacecraft/(i^2);
        I = phi; % Radiant Intensity source light at wavelenght X, W/sr
        receivedFlux = I*omega; % Irradiance in Watt/m^2 bit
        receivedFluxSun = irradianceSun; % Irradiance in Watt/m^2 bit

        radianceBit = (receivedFlux+receivedFluxSun)*reflectivityMarker; %W/m^2/sr
        radianceSc = (I*omegaSc+irradianceSun)*reflectivitySpacecraft; %W/m^2/sr

        signalSpacecraft(i) = radianceSc*areaSpacecraft*srCamera/(scPixels(i)^2); % W/m^2/pixel
        signalBit(i) = radianceBit*areaBit*srCamera/(bitPixels(i)^2); % W/m^2/pixel

        powerRatio(i) = signalBit(i)/signalSpacecraft(i);

        fluxDensityGamma = signalBit(i)*pixelSize^2*lambda/(planck*c); % Flux density in quantum units (photons/s) per pixel

```

```
89          S(i) = fluxDensityGamma*exposureTime*Qe*throughput; % Signal on detector in electrons per pixel
90
91          %fluxDensityGammaFull = (signalBit(i)+signalSpacecraft(i))*pixelSize^2*lambda/(planck*c); % Flux density in
                quantum units (photons/s/pixel)
92          %fullS(i) = fluxDensityGammaFull*exposureTime*Qe*throughput; % Signal on detector in electrons per pixel
93
94          % Detector Noise
95          shotNoise(i) = sqrt(S(i)); % Shot noise. For now we assume the system is shot noise limited
96          darkNoise = 28.2; % assuming prosilica datasheet is 28.2 electron/pixel
97          totalNoise(i) = shotNoise(i)+darkNoise;
98
99          snratio(i,z) = S(i)/totalNoise(i);
100         z = z+1;
101     end
102 end
```

List of References

[1] J.-C. Liou and N. L. Johnson. Risks in space from orbiting debris. *Science*, 311(5759):340–341, 2006. ISSN 0036-8075. doi: 10.1126/science.1121337.

[2] Minghe Shan, Jian Guo, and Eberhard Gill. Review and comparison of active space debris capturing and removal methods. *Progress in Aerospace Sciences*, 80:18 – 32, 2016. ISSN 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2015.11.001.

[3] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pages 85–94. IEEE, 1999.

[4] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[5] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[6] M. Fiala. Artag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 590–596 vol. 2, 2005.

[7] M. Fiala. Artag fiducial marker system applied to vision based spacecraft docking. In *Proc. Intl. Conf. Intelligent Robots and Systems (IROS) 2005 Workshop on Robot Vision for Space Applications*, pages 35–40, 2005.

[8] United Nations Office for Outer Space Affairs. Space debris mitigation guidelines of the committee on the peaceful uses of outer space. Technical report, 2010.

[9] Donald J Kessler and Burton G Cour-Palais. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research: Space Physics*, 83(A6):2637–2646, 1978.

[10] Donald J Kessler, Nicholas L Johnson, JC Liou, and Mark Matney. The kessler syndrome: implications to future space operations. *Advances in the Astronautical Sciences*, 137(8):2010, 2010.

[11] Heiner Klinkrad. Space debris. In *Encyclopedia of Aerospace Engineering*. American Cancer Society, 2010. ISBN 9780470686652. doi: 10.1002/9780470686652.eae325.

[12] United Nations Office for Outer Space Affairs. Space debris mitigation guidelines of the committee on the peaceful uses of outer space, 2010.

[13] European Cooperation for Space Standardization. Ecss-u-as-10c rev.1: Space sustainability adoption notice of iso 24113: Space systems - space debris mitigation requirements, 2019.

[14] J.-C. Liou. An active debris removal parametric study for leo environment remediation. *Advances in Space Research*, 47(11):1865 – 1876, 2011. ISSN 0273-1177. doi: https://doi.org/10.1016/j.asr.2011.02.003.

[15] B. Bastida Virgili and H. Krag. Strategies for Active Removal in LEO. In *Fifth European Conference on Space Debris*, volume 672 of *ESA Special Publication*, page 53, March 2009.

[16] C Priyant Mark and Surekha Kamath. Review of active space debris removal methods. *Space Policy*, 47: 194–206, 2019.

[17] Joseph N Pelton. A global fund for space debris remediation: A new way forward to address the mounting space debris problem. In *International Space University Symposium*, 2012.

[18] Joseph N. Pelton. *New Solutions for the Space Debris Problem*, pages 1–9. Springer International Publishing, Cham, 2015. ISBN 978-3-319-17151-7. doi: 10.1007/978-3-319-17151-7_1.

[19] David C Woffinden and David K Geller. Navigating the road to autonomous orbital rendezvous. *Journal of Spacecraft and Rockets*, 44(4):898–909, 2007.

[20] Antoine Petit, Eric Marchand, and Keyvan Kanani. Vision-based space autonomous rendezvous: A case study. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 619–624. IEEE, 2011.

[21] David Rijlaarsdam. Literature study relative navigation fordebris removal (render) - designing a fiducial marker system, 2020. Delft University of Technology.

[22] Masaaki Mokuno and Isao Kawano. In-orbit demonstration of an optical navigation system for autonomous rendezvous docking. *Journal of Spacecraft and Rockets*, 48(6):1046–1054, 2011.

[23] Daniel P Goodwin, Laura E Hembree, Joseph P Curran, David S Moyer, Russell L Strachan, Ian Mills, and Jean-Sebastian Valois. Orbiter space vision system on space shuttle flight sts-80. In *Visual Information Processing VI*, volume 3074, pages 18–28. International Society for Optics and Photonics, 1997.

[24] Timothy E. Rumford. Demonstration of autonomous rendezvous technology (DART) project summary. In Peter Tchoryk Jr. and James Shoemaker, editors, *Space Systems Technology and Operations*, volume 5088, pages 10 – 19. International Society for Optics and Photonics, SPIE, 2003. doi: 10.1117/12.498811.

[25] Domenico Accardo, Giancarmine Fasano, and Michele Grassi. *Vision Based Relative Navigation*, pages 267–305. Springer New York, New York, NY, 2013. ISBN 978-1-4614-4541-8.

[26] T Weismuller and M Leinz. Gn&c technology demonstrated by the orbital express autonomous rendezvous and capture sensor system. In *29th ANNUAL AAS GUIDANCE AND CONTROL CONFERENCE*. American Astronautical Society, 2006.

[27] Lorenzo Pasqualetto Cassinis, Robert Fonod, and Eberhard Gill. Review of the robustness and applicability of monocular pose estimation systems for relative navigation with an uncooperative spacecraft. *Progress in Aerospace Sciences*, 110:100548, 2019.

[28] Staffan Persson, Bjorn Jacobsson, and Eberhard Gill. Prisma- demonstration mission for advanced rendezvous and formation flying technologies and sensors. 2005. 56 th International Astronautical Congress.

[29] Jiří Šilha, Jean-Noël Pittet, Michal Hamara, and Thomas Schildknecht. Apparent rotation properties of space debris extracted from photometric measurements. *Advances in Space Research*, 61(3):844–861, 2018. ISSN 0273-1177. doi: https://doi.org/10.1016/j.asr.2017.10.048.

[30] Daniel Kucharski, Georg Kirchner, Franz Koidl, Cunbo Fan, Randall Carman, Christopher Moore, Andriy Dmytrotsa, Martin Ploner, Giuseppe Bianco, Mikhailo Medvedskij, et al. Attitude and spin period of space debris envisat measured by satellite laser ranging. *IEEE Transactions on Geoscience and Remote Sensing*, 52(12):7651–7657, 2014.

[31] J.A.F. Deloo and E. Mooij. Active debris removal: Aspects of trajectories, communication and illumination during final approach. *Acta Astronautica*, 117:277 – 295, 2015. ISSN 0094-5765.

[32] Roberto Opromolla, Giancarmine Fasano, Giancarlo Rufino, and Michele Grassi. A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Progress in Aerospace Sciences*, 93:53 – 72, 2017. ISSN 0376-0421.

[33] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[34] OpenCV. Camera Calibration and 3D Reconstruction. `https://docs.opencv.org/master/d9/d0c/group__calib3d.html`, 2021. Online. Accessed on 18/01/2021.

[35] Bert M Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International journal of computer vision*, 13(3):331–356, 1994.

[36] Sumant Sharma et al. Comparative assessment of techniques for initial pose estimation using monocular vision. *Acta Astronautica*, 123:435–445, 2016.

[37] Daniel F DeMenthon and Larry S Davis. Model-based object pose in 25 lines of code. *International journal of computer vision*, 15(1-2):123–141, 1995.

[38] Philip David, Daniel Dementhon, Ramani Duraiswami, and Hanan Samet. Softposit: Simultaneous pose and correspondence determination. *International Journal of Computer Vision*, 59(3):259–284, 2004.

[39] Steven Gold, Anand Rangarajan, Chien-Ping Lu, Suguna Pappu, and Eric Mjolsness. New algorithms for 2d and 3d point matching: pose estimation and correspondence. *Pattern Recognition*, 31(8):1019–1031, 1998. ISSN 0031-3203. doi: https://doi.org/10.1016/S0031-3203(98)80010-1.

[40] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (dls) method for pnp. In *2011 International Conference on Computer Vision*, pages 383–390. IEEE, 2011.

[41] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

[42] Denis Oberkampf, Daniel F DeMenthon, and Larry S Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, 1996.

[43] Gerald Schweighofer and Axel Pinz. Robust pose estimation from a planar target. *IEEE transactions on pattern analysis and machine intelligence*, 28(12):2024–2030, 2006.

[44] C.-P. Lu, G.D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000. doi: 10.1109/34.862199.

[45] Yuko Uematsu and Hideo Saito. Improvement of accuracy for 2d marker-based tracking using particle filter. In *17th International Conference on Artificial Reality and Telexistence (ICAT 2007)*, pages 183–189, 2007. doi: 10.1109/ICAT.2007.16.

[46] Daniel F Abawi, Joachim Bienwald, and Ralf Dorner. Accuracy in optical tracking with fiducial markers: an accuracy function for artoolkit. In *Third IEEE and ACM International symposium on mixed and augmented reality*, pages 260–261. IEEE, 2004.

[47] Michel Bondy, Rubakumar Krishnasamy, Derry Crymble, and Piotr Jasiobedzki. *Space Vision Marker System (SVMS)*. 2007. doi: 10.2514/6.2007-6185. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2007-6185`.

[48] James R Wertz, David F Everett, and Jeffery J Puschell. *Space mission engineering: the new SMAD*. Microcosm Press, 2011.

[49] Wiley J Larson, Douglas Kirkpatrick, Jerry Jon Sellers, L Dale Thomas, and Dinesh Verma. *Applied space systems engineering*. Mc Graw Hill, 2009.

[50] European Cooperation for Space Standardization. Ecss-e-st-10c rev.1: Space engineering - system engineering general requirements. ESTEC, Noordwijk, The Netherlands, 2017.

[51] Cecilia Haskins, Kevin Forsberg, Michael Krueger, D Walden, and D Hamelin. Systems engineering handbook. In *INCOSE*, volume 9, pages 13–16, 2006.

[52] European Cooperation for Space Standardization. Ecss-e-st-10-06c: Space engineering - technical requirements specification, 2009.

[53] P Colmenarejo, G Binet, L Strippoli, TV Peters, and M Graziano. Gnc aspects for active debris removal. *CEAS EuroGNC. Delft, The Netherlands*, 2013.

[54] Simone D'Amico, Mathias Benn, and John L Jørgensen. Pose estimation of an uncooperative spacecraft from actual space imagery. *International Journal of Space Science and Engineering 5*, 2(2):171–189, 2014.

[55] John Leif Jørgensen and Carl Christian Liebe. The advanced stellar compass, development and operations. *Acta Astronautica*, 39(9):775–783, 1996. ISSN 0094-5765. IAA International Symposium on Small Satellites for Earth Observation.

[56] M Delpech, JC Berges, S Djalal, and J Christy. Vision based rendezvous experiment performed during the prisma extended mission. In *Proceedings of the 23rd International Symposium on Space Flight Dynamics, Jet Propulsion Laboratory, Pasadena, California, USA*, 2012.

[57] Angelo V Arecchi, R John Koshel, and Tahar Messadi. Field guide to illumination. SPIE, 2007.

[58] Phillip C. Calhoun and Richard Dabney. Solution to the problem of determining the relative 6 DOF state for spacecraft automated rendezvous and docking. In Walter J. Fowski and Morris M. Birnbaum, editors, *Space Guidance, Control, and Tracking II*, volume 2466, pages 175 – 184. International Society for Optics and Photonics, SPIE, 1995. doi: 10.1117/12.211505. URL `https://doi.org/10.1117/12.211505`.

[59] Andrew Heaton, Richard Howard, and Robin Pinson. Orbital express avgs validation and calibration for automated rendezvous. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2008. doi: 10.2514/6.2008-6937.

[60] Hideyuki Tanaka, Yasushi Sumi, and Yoshio Matsumoto. A solution to pose ambiguity of visual markers using moire patterns. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3129–3134. IEEE, 2014.

[61] Richard T. Howard, Thomas C. Bryan, and Michael L. Book. On-orbit testing of the video guidance sensor. In Gary W. Kamerman and Christian Werner, editors, *Laser Radar Technology and Applications IV*, volume 3707, pages 290 – 300. International Society for Optics and Photonics, SPIE, 1999.

[62] John Christian, Heather Hinkel, Sean Maguire, Christopher D'Souza, and Mogi Patangan. The sensor test for orion relnav risk mitigation (storrm) development test objective. In *AIAA Guidance, Navigation, and Control Conference*, 2011.

[63] John A Christian and Scott Cryan. A survey of lidar technology and its use in spacecraft relative navigation. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, page 4641, 2013.

[64] Richard T Howard, Albert S Johnston, Thomas C Bryan, and Michael L Book. Advanced video guidance sensor (avgs) development testing. In *Spaceborne Sensors*, volume 5418, pages 50–60. International Society for Optics and Photonics, 2004.

[65] Francois Blais, J Angelo Beraldin, Luc Cournoyer, Iain Christie, R Serafini, K Mason, S McCarthy, and C Goodall. Integration of a tracking laser range camera with the photogrammetry-based space vision system. In *Acquisition, Tracking, and Pointing XIV*, volume 4025, pages 219–228. International Society for Optics and Photonics, 2000.

[66] H. F. L. Pinkney, C. I. Perratt, and S. G. MacLean. CANEX-2 space vision system experiments for Shuttle Flight STS-54. In Armin Gruen and Emmanuel P. Baltsavias, editors, *Close-Range Photogrammetry Meets Machine Vision*, volume 1395, pages 357 – 364. International Society for Optics and Photonics, SPIE, 1990. doi: 10.1117/12.2294291.

[67] Alfred M Bruckstein, Robert J Holt, Thomas S Huang, and Arun N Netravali. New devices for 3d pose estimation: Mantis eyes, agam paintings, sundials, and other space fiducials. *International Journal of Computer Vision*, 39(2):131–139, 2000.

[68] Thomas L. Saaty. How to make a decision: The analytic hierarchy process. *European Journal of Operational Research*, 48(1):9–26, 1990. ISSN 0377-2217. Desicion making by the analytic hierarchy process: Theory and applications.

[69] Thomas L Saaty. Decision making with the analytic hierarchy process. *International journal of services sciences*, 1(1):83–98, 2008.

[70] Thomas L Saaty and Liem T Tran. On the invalidity of fuzzifying numerical judgments in the analytic hierarchy process. *Mathematical and Computer Modelling*, 46(7-8):962–975, 2007.

[71] Valerie Belton and Tony Gear. On a short-coming of saaty's method of analytic hierarchies. *Omega*, 11 (3):228–230, 1983. ISSN 0305-0483.

[72] Thomas L Saaty. *Fundamentals of decision making and priority theory with the analytic hierarchy process*, volume 6. RWS publications, 2007.

[73] Evangelos Triantaphyllou and Stuart H Mann. Using the analytic hierarchy process for decision making in engineering applications: some challenges. *International journal of industrial engineering: applications and practice*, 2(1):35–44, 1995.

[74] Daniel Abawi, Joachim Bienwald, and Ralf Dörner. Accuracy in optical tracking with fiducial markers: An accuracy function for artoolkit. pages 260–261, 01 2004. doi: 10.1109/ISMAR.2004.8.

[75] AK Sharma and N Sridhara. Degradation of thermal control materials under a simulated radiative space environment. *Advances in space research*, 50(10):1411–1424, 2012.

[76] Joyce Dever, Bruce Banks, Kim de Groh, and Sharon Miller. Chapter 23 - degradation of spacecraft materials. In Myer Kutz, editor, *Handbook of Environmental Degradation of Materials*, pages 465–501. William Andrew Publishing, Norwich, NY, 2005. ISBN 978-0-8155-1500-5. doi: https://doi.org/10.1016/B978-081551500-5.50025-2.

[77] Evangelos Triantaphyllou and Stuart H. Mann. A computational evaluation of the original and revised analytic hierarchy process. *Computers & Industrial Engineering*, 26(3):609–618, 1994. ISSN 0360-8352. doi: https://doi.org/10.1016/0360-8352(94)90054-X.

[78] Evangelos Triantaphyllou and Alfonso Sánchez. A sensitivity analysis approach for some deterministic multi-criteria decision-making methods. *Decision sciences*, 28(1):151–194, 1997.

[79] Alfred M Bruckstein, Tuvi Etzion, Raja Giryes, Noam Gordon, Robert J Holt, and Doron Shuldiner. Simple and robust binary self-location patterns. *IEEE transactions on information theory*, 58(7):4884–4889, 2012.

[80] Bergkvist. Device for optic, preferably visual determination of a certain plane, U.S. Patent 4166699, 1979.

[81] B Kunkel. Sensing apparatus for determining the relative position between two bodies with plural emitters and a shading member, U.S. Patent 4710620, 1987.

[82] H. Tanaka, Y. Sumi, and Y. Matsumoto. Avisual marker for precise pose estimation based on lenticular lenses. In *2012 IEEE International Conference on Robotics and Automation*, pages 5222–5227, 2012.

[83] H. Tanaka, K. Ogata, and Y. Matsumoto. Solving pose ambiguity of planar visual marker by wavelike two-tone patterns. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 568–573, 2017.

[84] Hideyuki Tanaka, Yasushi Sumi, and Yoshio Matsumoto. A high-accuracy visual marker based on a microlens array. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4192–4197. IEEE, 2012.

[85] Hideyuki Tanaka, Yasushi Sumi, and Yoshio Matsumoto. Further stabilization of a microlens-array-based fiducial marker. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 297–298, 2013. doi: 10.1109/ISMAR.2013.6671813.

[86] I. Schillebeeckx, J. Little, B. Kelly, and R. Pless. The geometry of colorful, lenticular fiducial markers. In *2015 International Conference on 3D Vision*, pages 487–499, 2015.

[87] Ian Schillebeeckx and Robert Pless. Structured light field design for correspondence free rotation estimation. In *2015 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10, 2015. doi: 10.1109/ICCPHOT.2015.7168376.

[88] E. J. Larsen. Tracking head position and orientation, US Patent 8922644, 2014.

[89] Hong Xuan, Ian Schillebeeckx, and Robert Pless. Self calibrating lenticular markers and articulated pose estimation. In *2017 2nd International Conference on Robotics and Automation Engineering (ICRAE)*, pages 384–388, 2017. doi: 10.1109/ICRAE.2017.8291415.

[90] Ian Schillebeeckx and Robert Pless. Pose hashing with microlens arrays. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 600–614, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46487-9.

[91] J Zeil and M Al-Mutairi. The variation of resolution and of ommatidial dimensions in the compound eyes of the fiddler crab Uca lactea annulipes (Ocypodidae, Brachyura, Decapoda). *Journal of Experimental Biology*, 199(7):1569–1577, 07 1996. ISSN 0022-0949. doi: 10.1242/jeb.199.7.1569.

[92] Jerry E LeCroy, Dean S Hallmark, and Richard T Howard. Effects of optical artifacts in a laser-based spacecraft navigation sensor. In *Sensors and Systems for Space Applications*, volume 6555, page 655507. International Society for Optics and Photonics, 2007.

[93] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[94] Allied Vision. Prosilica gt 4096 nir datasheet version 1.1.4. `https://www.alliedvision.com/en/products/cameras/detail/Prosilica%20GT/4096NIR/action/pdf.html`, 2021. Online. Accessed on 20/03/2021.

[95] Alan B. Marchant, Kyle D. Jeppson, and Ryan T. Scott. Conspicuity tape for enhanced laser range finding. *Optical Engineering*, 49(4):1 – 7, 2010.

[96] Richard T. Howard, Helen Johnson Cole, John Larkin Jackson, Gary W. Kamerman, and Donald K. Fronek. Automatic rendezvous and docking system test and evaluation. In Gary W. Kamerman, editor, *Laser Radar Technology and Applications II*, volume 3065, pages 131 – 139. International Society for Optics and Photonics, SPIE, 1997. doi: 10.1117/12.281002. URL `https://doi.org/10.1117/12.281002`.

[97] Ernest Hilsenrath, Howard H. Herzig, Donald E. Williams, Carol J. Bruegge, and Albert E. Stiegman. Effects of Space Shuttle flight on the reflectance characteristics of diffusers in the near-infrared, visible, and ultraviolet regions. *Optical Engineering*, 33(11):3675 – 3682, 1994. doi: 10.1117/12.181932. URL `https://doi.org/10.1117/12.181932`.

[98] Y Salomon, N Sternberg, I Gouzman, G Lempert, E Grossman, D Katsir, R Cotostiano, and T Minton. Qualification of acktar black coatings for space application. In *Proceedings of the International Symposium on Materials in a Space Environment, Aix-En-Provence, France*, 2009.

[99] Sharon KR Miller and Bruce Banks. Degradation of spacecraft materials in the space environment. *MRS bulletin*, 35(1):20–24, 2010.

[100] Meike List, Stefanie Bremer, Benny Rievers, and Hanns Selig. Modelling of solar radiation pressure effects: Parameter analysis for the microscope mission. *International Journal of Aerospace Engineering*, 2015, 2015.

[101] A. Paillous and C. Pailler. Degradation of multiply polymer-matrix composites induced by space environment. *Composites*, 25(4):287–295, 1994. ISSN 0010-4361. doi: https://doi.org/10.1016/0010-4361(94)90221-6.

[102] Martin Zwick, Irene Huertas, Levin Gerdes, and Guillermo Ortega. Orgl–esa's test facility for approach and contact operations in orbital and planetary environments. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2018.

[103] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[104] John Wang and Edwin Olson. Apriltag 2: Efficient and robust fiducial detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4193–4198. IEEE, 2016.

[105] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[106] Nathan Pierre Deom, Olivier Bruls, and Thierry Jacques. Programming by demonstration using fiducial markers. In *Proceedings of the Robotix Academy Conference for Industrial Robotics (RACIR) 2019*, 2019.