University of Tartu

Faculty of Science and Technology

Institute of Technology

Sandra Schumann

Using Star Identification Algorithms on ESTCube-2 Star Tracker

Master's Thesis (30 ECTS)
Robotics and Computer Engineering

Supervisors:

MSc Hendrik Ehrpais
MSc Tõnis Eenmäe

Tartu 2019

Resümee

Tähtede identifitseerimisalgoritmide kasutamine ESTCube-2 tähejälgimiskaameral

Käesolev töö hindab ESTCube-2 tudengisatelliidi tähejälgimiskaamera teostatavust ja tuvastab selle oodatavad jõudlusnäitajad. Mõõdetakse piirtähesuurus tähejälgimiskaamera riistvara jaoks ja geomeetrilise modelleerimise abil hinnatakse signaal-müra suhte vähenemist satelliidi pöördliikumise tõttu. Leitud piirtähesuurust kasutatakse tähtede identifitseerimiseks kasutatavate kujundituvastusalgoritmide optimaalsete parameetrite hindamiseks. Töö arendab välja ka viisi võrdlusandmebaasi loomiseks ja efektiivseks struktureerimiseks.

Katsetulemused näitavad, et kasutades leitud parameetreid, on tähejälgimiskaamera võimeline määrama satelliidi asendi väikeste pöörlemiskiiruste (kuni 5 kraadi sekundis) korral. Tulemustest nähtub ka seadme võime määrata asendit kuni 10 korda sekundis. Satelliidi missioon sisaldab ka kiire pöörlemise faasi, mille jooksul tähejälgimiskaamera enam igal ajahetkel täpset asendihinnangut leida ei suuda. Samas suudab tähejälgimiskaamera töös soovitatud algoritmi seadistusi kasutades ka 5 kraadist sekundis kiiremate pöörlemiskiiruste juures endiselt pakkuda asendi- ja orbiidikontrolli alamsüsteemile väärtuslikku teavet. Seega võib ESTCube-2 tähejälgimiskaamera antud olukorras kaasaegsetest kommertslahendustest võimekamaks osutuda.

CERCS: T320 Kosmosetehnoloogia; T111 Pilditehnika; P520 Astronoomia, kosmoseuuringud, kosmosekeemia

Märksõnad: asendituvastus, nanosatelliidid, tähejälgimiskaamera, algoritmid

Abstract

Using Star Identification Algorithms on ESTCube-2 Star Tracker

This thesis estimates the feasibility and determines the expected performance characteristics of a star tracker for ESTCube-2 student satellite. It measures limiting magnitude for the ESTCube-2 star tracker hardware and estimates the decrease in signal-to-noise ratio due to the spacecraft's rotational motion using geometric modelling. The acquired limiting magnitude is used to determine the optimal parameters to use with the pattern recognition algorithms for star identification purposes. The work also develops a way of creating and structuring a reference database in an effective way.

Test results indicate that using the acquired parameters, the star tracker will be able to determine spacecraft's attitude for cases of slow rotation up to 5 degrees per second. Results also indicate capability of attitude determination up to 10 times per second. The spacecraft's mission also contains a phase of fast rotation, during which the star tracker will no longer be able to produce accurate attitude estimates at all times. With the algorithm configurations recommended in the thesis the ESTCube-2 star tracker could still provide a valuable contribution to the attitude and orbital control subsystem while experiencing angular velocities greater than 5 degrees per second. Thus it may outperform state-of-the-art commercial nanosatellite star trackers in that particular situation.

CERCS: T320 Space technology; T111 Imaging, image processing; P520 Astronomy, space research, cosmic chemistry

Keywords: attitude determination, nanosatellites, star tracker, algorithms

Contents

List of Figures

List of Tables

Abbreviations, Constants, Generic Terms

4-connectedness - the concept of defining neighbouring pixels in a two-dimensional square lattice by pixels that share an edge (as opposed to those that share an edge or a corner)

AOCS - attitude and orbital control subsystem

Attitude - the orientation of a spacecraft determined by the relationship between its axes and some defined reference

CMOS - complementary metaloxidesemiconductor, a technology for constructing integrated circuits

CPU - central processing unit

Declination - the angular distance of a given point from the celestial equator as measured along a great circle passing through the celestial poles (comparable to latitude)

FoV - field of view, the open observable area an instrument can see

FPGA - field programmable gate array, an integrated circuit designed to be configurable after manufacturing

FWHM - full width at half maximum, a way of measuring the width of a function between the points at which the dependent variable is equal to half of its maximum value

Ground station - a radio station on Earth for communication with spacecraft

Hash - output of a hash function

Hash function - a map from some input value to output of a certain length

k**-d tree** - k-dimensional tree, a data structure used for organising points in k-dimensional space

Limiting magnitude - faintest apparent magnitude of a celestial body that is detectable by some given instrument

Lost-in-space problem - the problem of determining a spacecraft's attitude without any *a priori* knowledge of it

Magnitude - a number representing the brightness of a celestial body, measured on a logarithmic scale in which an increase of five units corresponds to a reduction in the brightness of light by a factor of 100

MCU - microcontroller unit, a small computer in an integrated circuit

Monte Carlo methods - a class of algorithms that obtain numerical results via random sampling

Nadir - the point of the celestial sphere that is vertically below the observer

Nadir-pointing - the action of maintaining pointing direction directly below the satellite perpendicular to Earth

Quantum efficiency - the ratio of the number of photons detected in a photosensitive device to the number of photons falling onto the device for a specific wavelength

Right ascension - the angular distance along the celestial equator between the vernal equinox and the point of intersection between the hour circle through a given body and the celestial equator, measured eastward (comparable to longitude)

Signal-to-noise ratio - a measure comparing the level of a desired signal with the background noise level

SIMBAD - the Set of Identifications, Measurements and Bibliography for Astronomical Data, an astronomical database, `http://simbad.u-strasbg.fr/`

Star tracker - a celestial reference device that recognizes star patterns and determines its attitude based on them

Zenith - the point of the celestial sphere that is vertically above the observer

1 Introduction

ESTCube-2[1] is a student satellite that aims to test several experiments, among them a new type of electric solar sail. In order to be able to take photos of objects of interest, use fast communication with the ground station and deploy the solar sail, the satellite needs to have an accurate estimate of its attitude (orientation). The satellite has several sensors for attitude determination: magnetometers (measuring magnetic field), a Sun sensor (measures the location of the Sun) and a star tracker, the latter being the focus of this thesis [2, 3].

The basic functionality of a star tracker relies on the idea of celestial navigation that has been used by humans since ancient times [4]. Star tracker's camera takes a picture of the sky and stars are identified on the picture. The patterns that appear on the picture are matched to a database. This way stars on the picture are identified as specific stars in the sky. Based on known locations of these stars the star tracker now has information regarding which direction its camera is pointed at and can now calculate spacecraft's attitude.

The advantage of using a star tracker as an attitude determination method stems from its accuracy. While the satellite also relies on other sensors capable of attitude determination, a star tracker is able to provide more precise estimates of its current attitude than any of the alternative methods [5]. On the other hand, star trackers also have a downside of solving a computationally difficult problem. As such, the star tracker on ESTCube-2 cannot update its estimate as fast as other sensors on board. Star tracker also fails when the Earth blocks its field of view or when the Sun overexposes the entire picture. Lastly, it can give false estimates when looking at areas of sky with few sufficiently bright stars.

Another important aspect of using a star tracker on ESTCube-2 relates to the fact that unlike magnetometers, it is still capable of giving an estimate at a significant distance from Earth and its magnetic field. It also provides a way of attitude determination when Sun leaves the field of view or the satellite is eclipsed, thus mitigating problems that arise for Sun sensors. While ESTCube-2 will not leave Earth's orbit, testing out star tracker's functionality is essential for future missions, such as the ESTCube-3 nanosatellite that is planned to operate in the lunar orbit [3].

1.1 Problem Overview

Software running on ESTCube-2 essentially needs to solve three problems: first it needs to detect stars on a picture taken of the sky, secondly it has to identify these stars and lastly calculate the spacecraft's attitude based on the identified stars. While solutions for the third task have been established for a long time [6], achieving a good performance relies largely on the

[1] `https://www.estcube.eu/en/estcube-2`

algorithmic choices made for the first two tasks.

While star trackers have been in use for a long time, building a functional one for a nanosatellite still imposes a challenge due to constraints set by the small form factor. Additionally, requirements for ESTCube-2 become even more complex due to the fact that the satellite carries an electric solar sail experiment. The satellite will need to spin fast in order to deploy the sail. This significantly complicates the star tracker's operations due to image distortions caused by the rotational movement.

Taking all into account, this thesis attempts to answer the following questions.

1. Which algorithms would be the best for ESTCube-2 star detection and matching purposes?
2. What are the expected performance characteristics of the star tracker under the conditions of various angular velocities?
3. For which angular velocities is the star tracker expected to work at all?

1.2 Thesis Organization

This thesis is largely analytical with a heavy simulation component.

Chapter 2 describes previous work done on the subject matter, both algorithms implemented for other star trackers as well as hardware and firmware implementations of ESTCube-2 star tracker. Chapter 3 determines the magnitude of faintest stars visible to the camera under flight conditions. Chapter 4 uses results obtained in Chapter 3 to simulate matching algorithms and provides an estimate of expected performance characteristics.

This work briefly introduces, but does not cover thoroughly ESTCube-2 star tracker hardware. It also does not cover firmware necessary for reading data from the camera sensor or implement algorithms on the star tracker FPGA. While all of these components are necessary for operating the star tracker, hardware and firmware implementations fall outside the domain of attitude determination algorithms based on star locations and this thesis. However, information regarding ESTCube-2 star tracker hardware is briefly covered in Section 2.3 since the hardware imposes certain restrictions on the software and algorithmic choices.

2 Previous Work

2.1 Astrometric Calibration

Astrometry is a specific branch of astronomy the task of which is to measure positions of stars and other celestial bodies. Astrometric calibration, specifically, refers to the procedure of creating a mapping between coordinates on an image and celestial coordinates. There are several different ways of creating this mapping. Choices regarding which method is most useful depend on the specific application.

A long-time standard for astrometric calibration algorithms uses the idea of geometric hashing [1, 7, 8]. The advantage of this method is its applicability in a wide range of scenarios: it can be used on images where the camera's properties (such as field of view) are unknown, as well as those where hardware specifications are well-established. It is applicable in situations where an initial estimate of a location exists, as well as those where it is completely unknown. Additionally, it has successfully been used by star trackers in space [9].

One example of a software that successfully uses the hash-based approach for star identification, even under the unfavourable conditions of lacking information about the camera's properties and any initial estimate of which area of the sky a picture might be taken of, is Astrometry.net [1]. This software has established itself as a very fast-performing option in the field of astronomy and thus is chosen as an example for illustrating the concept of geometric hashing in Section 2.1.1.

2.1.1 Astrometry.net

Creators of Astrometry.net[1], software for blind astrometric calibration of arbitrary astronomical images, solve the problem of identifying stars on arbitrary images by using geometric hashes and determining the celestial coordinates where the image has been taken [1]. Namely, after detecting stars on the images, they create groups of stars and obtain geometric hashes of each group. A hash in this context refers to a way of converting data into a representation of fixed size and form. These hashes are then compared to entries in a database. Database matches are checked for validity. In case of a sufficient match the midpoint of the field of view along with a list of some other parameters is calculated and returned to the user.

The hashing principle used by authors of Astrometry.net is illustrated for four stars on Figure 2.1. The authors find two stars that are the furthest apart on the image (call these two stars A and B), define a two-dimensional coordinate system such that A is located at $(0, 0)$ and B at $(1, 1)$ (the coordinate system is rotated with respect to the edges of the image such that the

[1] `http://astrometry.net/`

two axes would have the same scale in image pixels). Coordinates of the other two stars, C and D, are then used as the geometric hash for this group of four stars. In order to make sure that a particular group of stars generates the same hash regardless of its orientation on the image and choice of C and D, it is required that $x_C \leq x_D$ and $x_C + x_D \leq 1$ (where x_C and x_D are the x-coordinates of C and D, respectively). An analogous geometric hash has been defined for groups of size 3 or 5, with 2 or 6 coordinates making up a hash, respectively.

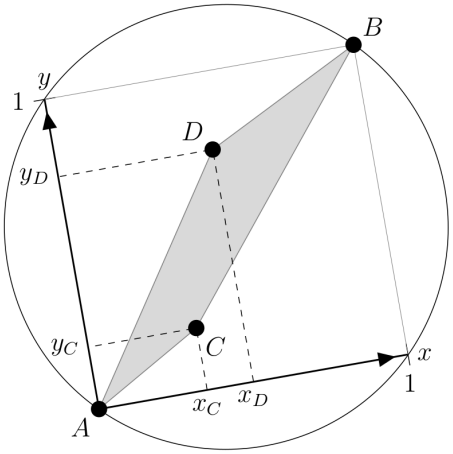


Figure 2.1: Hashing principle used by Astrometry.net [1].

While Astrometry.net's code is capable of using star groups of sizes 3, 4 and 5, authors default to groups of 4 [1]. They justify the choice by pointing out that decreasing the number of stars used in a group leads to more false positives due to star groups being less distinctive (when comparing groups of 3 to groups of 4, this means 2-dimensional hashes instead of 4-dimensional). However, increasing the group size leads to problems of likelihood of all stars from a group in a database simultaneously being in the field of view decreasing and the database size increasing significantly ($O(N^g)$ where N is the number of stars and g group size).

A clearly distinctive feature of Astrometry.net's problem is that since the authors try to solve the problem of astrometric calibration for any image, possibly without any reasonable *a priori* estimate of the area of sky displayed on the image and not even knowing the size of field of view, they use normalized hashes (that is, hashes are invariant under scaling in addition to rotation and translation) [1]. However, ESTCube-2 does not have this restriction, as the size of field of view is known beforehand. Therefore, it is beneficial for ESTCube-2 to use non-normalized hashes instead, as the extra information makes hashes more distinctive than using normalized hashes and thus reduces the chances of false positives.

Additionally, Astrometry.net's limitations on searching through its database are different from ESTCube-2's even when leaving aside the field of view uncertainty. While Astrometry.net often needs to search the entire database of hashes, ESTCube-2 can often receive an attitude estimate from other sensors and limit the search to the part of the database that corresponds to the received estimate. This increases search speed proportionally to the reduction in the number of database entries examined. However, Astrometry.net does not need to work in real time, while ESTCube-2 needs to obtain matches relatively fast while running on a low-powered MCU. These distinctions may end up being a key aspect for reasonable ESTCube-2 star tracker algorithmic architecture choices.

2.2 Star Trackers

The first practical use cases of star tracker technology were used in aircraft and missile guidance to improve estimates by other sensors [10]. In spacecraft applications star trackers have been used since the 1980s [11]. Several different types of star trackers exist, three main types of which are star scanners, gimbaled star trackers and fixed head star trackers. The first one uses the spacecraft's rotation to scan a view of the star field through a small slit, the second one searches reference stars by mechanically moving the detector, and the third one uses the pattern left by stars on a photosensitive array to calculate the spacecraft's attitude [10]. The rest of this thesis focuses exclusively on fixed head star trackers.

Typical star tracker accuracy is around the order of magnitude of 1 to 10 arcseconds for pitch/yaw and 10 to 100 arcseconds for roll [12–17]. To this day star trackers are only capable of operating under the conditions of low spacecraft angular velocity [12]. Maximum angular velocities handled by star trackers are on the order of $3 - 7$ degrees per second [13, 15]. Technologies that combine star trackers with gyroscopes advertise being able to bring that number up to 20 degrees per second [18]. The fact that typical star trackers can only handle relatively low angular velocities poses a problem for ESTCube-2, which will in fact be rotating rapidly during certain phases of the mission.

The main downsides of star trackers are their mass, power consumption and computational expense, all typically higher than for other sensors [10]. These disadvantages become glaringly apparent when a star tracker is used on small satellites. Designing a functional star tracker for microsatellites was a challenging task in 2006 [10] and during the next decade the task of making one work on a nanosatellite has also proven to be highly nontrivial. As an example, AeroAstro miniature star trackers on board the first two BRITE satellites, launched in 2013, failed to calculate attitude reliably when not directed at dense star fields [19].

There are few star trackers small enough to be used on nanosatellites that have also successfully been tested in space, of which one of the most well-known ones is ST-16RT2 by Sinclair Interplanetary [13, 20]. ST-16RT2 and the ST-200 star tracker by Hyperion Technologies are examined more closely as examples of state-of-the-art nanosatellite star trackers [17].

2.2.1 Hyperion Technologies and ST-200

The ST-200 star tracker developed by Hyperion Technologies is a fully autonomous, low power star tracker [17]. Its low mass of 42 g and nominal power consumption of 600 mW allow it to be used on nanosatellites. The device offers a $5Hz$ update rate, full lost-in-space solution, attitude determination accuracy of 30 arcseconds for pitch and yaw and 200 arcseconds for roll. The device is guaranteed to be functional for rotational movement up to 0.3 degrees per second for tilt and 0.6 degrees per second for roll. Unfortunately details regarding the precise implementation of ST-200 are not available to the general public (similarly to many other commercial star trackers) and therefore make it difficult to learn from their design choices when building a star tracker for ESTCube-2.

2.2.2 Sinclair Interplanetary and ST-16RT2

Star trackers by Sinclair Interplanetary[2] can be considered to be some of the most successful nanosatellite-compatible star trackers, as their functionality has repeatedly been demonstrated on spacecrafts such as SkySats 1 to 15, several BRITE-constellation nanosatellites, several Landmapper-BC satellites and others [21]. Their ST-16 star trackers were launched on board SkySay-1 in 2013 [9] and the revised version ST-16RT2 is at this time their main star tracker product [13, 21].

ST-16RT2 [13] weighs 158 g (without baffle) and has an average power consumption of under 500 mW. It offers accuracy of 5 arcseconds for pitch and yaw and 55 arcseconds for roll. The star tracker can handle rotational movement up to 3 degrees per second and provides full lost-in-space solution for each frame at $2Hz$ rates.

Sinclair Interplanetary has published a significant number of articles[3] detailing several of their design choices from both hardware as well as algorithmic perspective. They also describe the points of failure discovered during their first space flight in detail [9], which is extremely useful for projects such as ESTCube-2. Therefore the design of our satellite has been heavily influenced by work by Sinclair Interplanetary.

2.3 ESTCube-2 Hardware Specifications

ESTCube-2 electronics and optics have been designed independently of the work done by the author of this thesis. The star tracker hardware has been designed based on the ESEO secondary camera and lessons learned from the articles written by the Sinclair Interplanetary team. Choices made for hardware also affect the optimal choices for star detection and identification algorithms.

The star tracker captures images of the sky using a MT9P031 1/2.5-inch CMOS digital image sensor [22]. The sensor captures a 2592x1944 monochrome image. The sensor has 12-bit ADC resolution and an electronic rolling shutter.

In front of the sensor is a high-resolution CCTV lens by Marshall Electronics, Inc. [23]. The lens is designed for 1/3-inch image sensors, meaning that at the edges and corners of the 1/2.5-inch image sensor the image quality is not guaranteed to be optimal. The lens has a focal length of 16.0 mm, focal ratio of 1.2 and field of view of $17.2° \times 13°$.

Image is read from the sensor using a Cyclone IV EP4CE22 field-programmable gate array (FPGA) [24]. The FPGA processes the image and finds stellar images. It interacts with SDRAM [25], FRAM [26] and flash storage [27]. Operation of FPGA, sensor and FRAM storage is controlled by an ARM Cortex-M4 CPU [28]. The CPU also performs computations that are either lightweight enough or algorithmically too complicated for the FPGA.

[2] `http://www.sinclairinterplanetary.com/`
[3] `http://www.sinclairinterplanetary.com/publications`

Additionally, the star tracker has to take into account mechanical constraints. ESTCube-2 will be testing out an electric solar sail that will be unreeled from the satellite by spinning the spacecraft rapidly around a particular axis. The star tracker is mounted such that its optical axis is parallel to the axis of this fast rotation. Angular velocity for deployment of the electric solar sail can reach $360°s^{-1}$ at the beginning of deployment and is reduced to $20°s^{-1}$ as the sail is deployed [29]. Additionally when the satellite is nadir-pointing, it is subjected to rotational movement due to the satellite's orbital motion, causing a rotation of 360 degrees over a period of 97 minutes.

2.4 Star Detection on FPGA

One major subtask of creating a functional star tracker for ESTCube-2 is detecting locations of stellar images (henceforth called blobs because of their not necessarily symmetric shape) on a captured image. This detection is done using an FPGA. Ayal solves this problem in his 2016 bachelor's thesis [30]. The thesis implements an algorithm proposed by Lindh in 2014 [31] with improvements in the form of adaptive thresholding and more accurate sub-pixel level centroid detection.

Ayal's algorithm uses adaptive thresholding [30]. The algorithm continuously updates a running average pixel value μ and standard deviation σ as the stream of pixels is being read from the sensor. For every pixel, it compares the pixel value to the current estimate of $\mu + 5\sigma$. If the pixel value exceeds this threshold, it is considered to belong to an image of a star. The algorithm works based on the assumption that noise on the image follows a normal distribution and therefore any pixel value 5σ above the mean must come from a star. It also assumes that the amount of total brightness added to the picture by stars is negligible (so they do not affect mean or standard deviation significantly). His algorithm then continues by assigning all pixels brighter than the threshold value to blobs according to 4-connectedness.

Sinclair Interplanetary, however, describe a slightly different algorithm being used on their ST-16 star tracker [9]. The algorithm similarly thresholds the image to determine the pixels that might represent stars. However, instead of using a global adaptive threshold, it compares the brightness of a pixel to the average brightness of the pixels in some neighbourhood window, in their case 128×1 pixels. If the brightness of the pixel exceeds this average value by more than some certain constant value, it is considered to be a candidate pixel for an image of a star, called a *lit pixel*.

After all the lit pixels have been determined, they are grouped into contiguous blobs. Each blob is evaluated according to two criteria: number of pixels belonging to the blob and the sum of all pixel values belonging to the blob. If both values exceed certain thresholds, the blob is considered to be an image of a star. All the thresholds used by Sinclair Interplanetary were first assigned some values before the start of the flight. During its first flight the values were chosen to be large enough in order to avoid false positives, but later modified in-orbit, as the initial estimates did not give expected performance and were found to be too conservative [9].

With respect to Ayal's algorithm, Sinclair's approach has the advantage of having already been tested in space. Namely, Sinclair Interplanetary's star trackers have been used for pointing Sky-

box Imaging SkySat-1 and BRITE-constellation nanosatellites [9, 19]. However, three different thresholds used by Sinclair were correctly determined only after the satellite had already taken test images of the sky while in orbit and transmitted them to Earth. If ESTCube-2 is to use a similar approach to Sinclair Interplanetary, then setting these thresholds would need to be made as easy as possible by just sending a couple of new constants to the on-board computer system of the satellite. This approach also requires an option to download images obtained by the star tracker camera.

3 Star Detection and Limiting Magnitude

The optimal size of star groups that are used for matching images to the database depends heavily on the number of visible stars in the field of view at various points in time. An accurate estimation of this number can be made in three steps:

1. determining the faintest stars that are detected by the star tracker (henceforth called the *limiting magnitude*);
2. estimating how much the signal level of stellar images drops due to satellite's rotational motion (*smear*);
3. simulating looking in various directions of the sky and finding all stars in the FoV of the star tracker's camera (*static availability*).

3.1 Static Limiting Magnitude

In order to determine the limiting magnitude of ESTCube-2 star tracker camera, this thesis took test pictures for analysing purposes using the selected optics and CMOS sensor for ESTCube-2 star tracker camera. Camera electronics used were different from ESTCube-2 electronics due to its development not being finished at the time. Instead, the electronics prototype for European Student Earth Orbiter's secondary camera was used[1]. This substitution of electronics is not relevant to optical quality of the pictures taken. Details of the setup have been listed in Appendix A.

3.1.1 Limiting Magnitude Estimation Using Longer Exposure Times

When the instrument is in space, the star tracker camera will most likely be taking pictures using 0.1-second exposure time. However, for estimating limiting magnitude on 0.1-second exposure time pictures, test images taken using longer exposure times can be helpful. The amount of photons arriving from the stars that fall onto the sensor is proportional to the exposure time used when taking the picture. Therefore the total amount of signal collected by the sensor pixels is proportional to exposure time. This principle allows calculating limiting magnitude based on images of different exposure time and then compensating for the difference mathematically. Measuring limiting magnitude on pictures with longer exposure times provides a way of achieving a more accurate result, because on pictures with shorter exposure times the contribution from various sources of noise is rather high and therefore obtaining a precise measurement is difficult.

[1] https://www.esa.int/Education/ESEO/Micro_Cameras

More precisely, when comparing exposure times of 10 seconds to 0.1 seconds, the exposure time of 0.1 seconds would collect 100 times less photons from the signal sources (stars) than the 10-second exposure time. According to Pogson's ratio, given two signal sources with flux densities F_1 and F_2, the difference of their magnitudes m_1 and m_2 respectively is given by

$$\Delta m = -5 \log_{100} \frac{F_1}{F_2}$$

where $\Delta m = m_1 - m_2$. Thus, if the limiting magnitude detected on a 10-second exposure time picture is m_{10}, then the limiting magnitude for 0.1 seconds would be $m_{10} - 5$. Similarly, one can calculate the limiting magnitude for 0.1 seconds from the limiting magnitude of a 1-second exposure time image by subtracting 2.5 magnitudes.

3.1.2 Manual Estimation

The limiting magnitude can be estimated visually from the test pictures. For this, a test image was taken using an exposure time of 10 seconds in the direction of the Ursa Major constellation. The image was manually thresholded to bring out images of stars. Stars were then matched to those visible in the same region on Stellarium planetarium software[2] and magnitudes of the faintest stars were collected. Operating this way, the author determined the limiting magnitude to be 9.95, which can reasonably be rounded to 10 due to measurement errors and gives a visually verified ballpark estimation.

Estimation results mentioned in this section were then used as a sanity check for automated limiting magnitude detection. This is due to the fact that the manual estimation method does not provide a viable way for processing a larger number of frames. Limiting magnitude estimation was performed on a larger scale using semi-automated systems based on works by Ayal and Sinclair Interplanetary instead [9, 30].

3.1.3 Noise Characterization

In order to properly use automated algorithms for star detection and set necessary parameter values, it was necessary to characterize noise inherent in the system and seen on the frames. For this, 20 dark frames (i.e. frames taken with lens cap in front of the camera) were obtained for each combination of three different exposure times (10, 1.0 and 0.1 seconds) and three different gain values (1, 8 and 15.8). Of those, 60 dark frames obtained using the gain of 15.8 were analysed further in this section. The dark frames were obtained in 4 sets. Each set included 5 dark frames with each exposure time, giving 15 frames total. During analysing, it became apparent that the first frame of each set displayed anomalous noise behaviour. This can be attributed to the system behaving differently at startup. As during flight the camera is expected to be continuously operating at all times (with the exception of shutting down or rebooting the star tracker, if such need should arise), those frames were discarded in the following analysis, leaving 16 valid dark frames with the 10-second exposure time and 20 valid dark frames with exposure times of 1.0 and 0.1 seconds each.

[2] `https://stellarium.org/`

Valid dark frames were then analysed for noise distribution. Figure 3.1a shows how brightness of a certain pixel on one dark frame is related to the brightness of the same pixel on another dark frame. Pearson correlation coefficients r were calculated for each exposure time and can be seen on the graph. The correlation is strong ($r = 0.93$) for the longest exposure time ($t = 10s$) and medium to low ($r = 0.46$, $r = 0.40$) for others. This indicates that some pixels are more likely to show up brighter than others. Checking these pixels confirmed that the pixels in question are so-called *hot pixels* that generate more dark signal than typical ones. The correlation is lower for shorter exposure times, because random noise has a higher proportion in the signal gathered by the pixels.

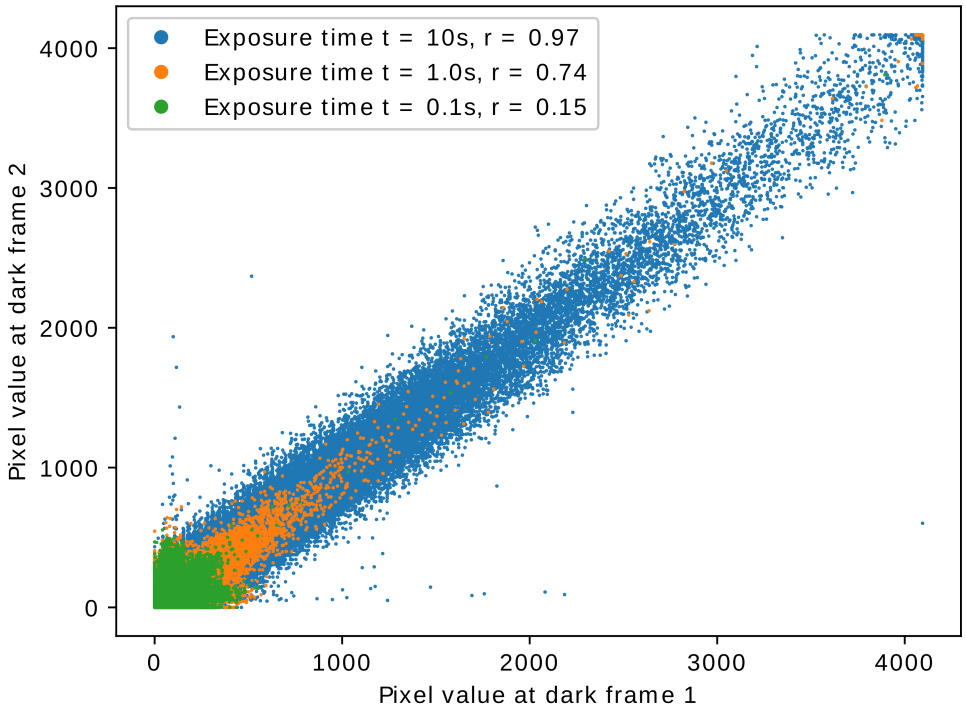
In order to correct for this bias, an average dark frame was computed for each exposure time using 14 dark frames. Two dark frames not used for computing the average dark frame were then taken, the average dark frame was subtracted from both (with negative resulting pixel values replaced by 0) and correlation between pixels in these two frames was then calculated. Results are shown on Figure 3.1b. Results show that after this procedure the pixel values only indicate very weak correlation between frames. Subtracting the average dark frame is also performed in subsequent analysis and should be done on the satellite when the star tracker is in operation.

Star detection methods rely on the fact that it is highly unlikely that several adjacent pixels could simultaneously give higher sensor readings simply due to noise. Because of this it is important that values of adjacent pixels not be correlated on a dark frame. For identifying stars as 4-connected blobs, it suffices to observe correlation between pixel values in vertical and horizontal directions. Figures 3.2a and 3.2b respectively display these results. The results show that the likelihood of a higher pixel value is not increased by higher neighbouring pixel values, neither vertically nor horizontally. Therefore it can be concluded that there is no such correlation, as the absolute value of Pearson correlation coefficient is around 0.1 or smaller.

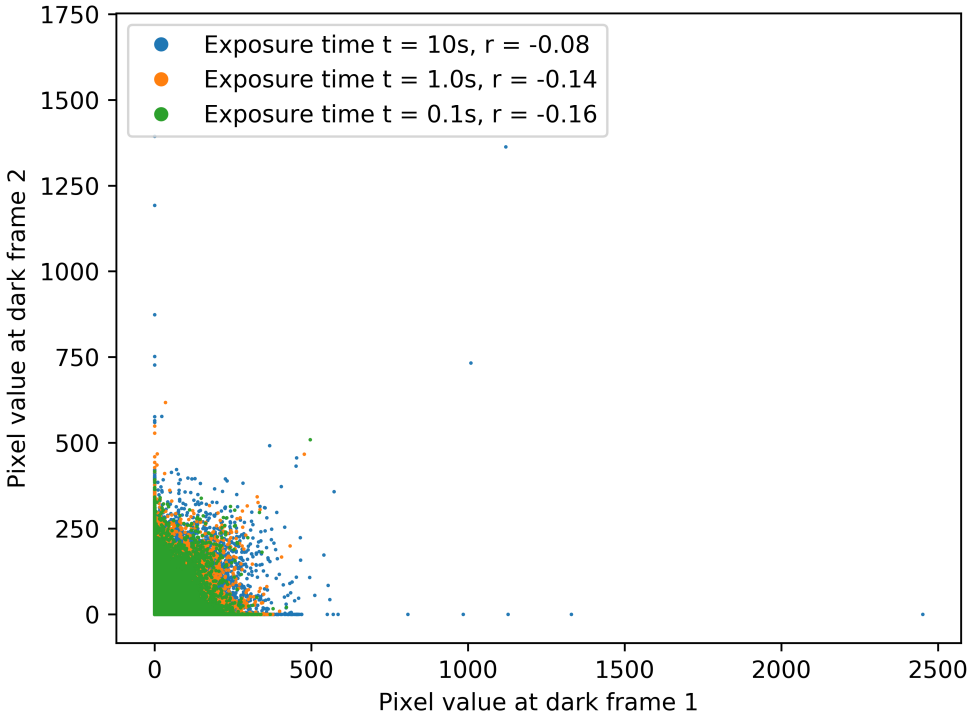
3.1.4 Setting Parameters for Automatic Star Recognition

Sinclair Interplanetary imposes the restriction that any cluster of pixels must have at least the size of 6 for it to be considered as a candidate for a star. Let the probability be p that a certain pixel value (after subtracting the average dark frame) crosses some threshold T, and let the probability of this happening be independent for all pixels. If this is the case, the probability of any 6 specific pixels crossing this threshold is p^6. This can be used to calculate the probability that a star spontaneously forms on the picture due to noise.

For this, it is necessary to know in how many different ways a cluster of 6 pixels that are in some way connected can be formed. This can be solved by constructing a grid of 11×11 pixels and choosing the pixel in the middle row and column as a pixel hypothetically being processed by the star detection algorithm. By looking at all combinations of 5 pixels out of the remaining 120 and counting how many of them give rise to configurations where all of those 5 are connected to the middle pixel in some way, one can determine the number of ways that a star can spontaneously form around a pixel being processed. The results show that given any pixel, there are 1296 distinct ways to construct a cluster of 6 pixels around this one such that the cluster would be 4-connected. Therefore, the probability of a cluster of size at least 6 forming a

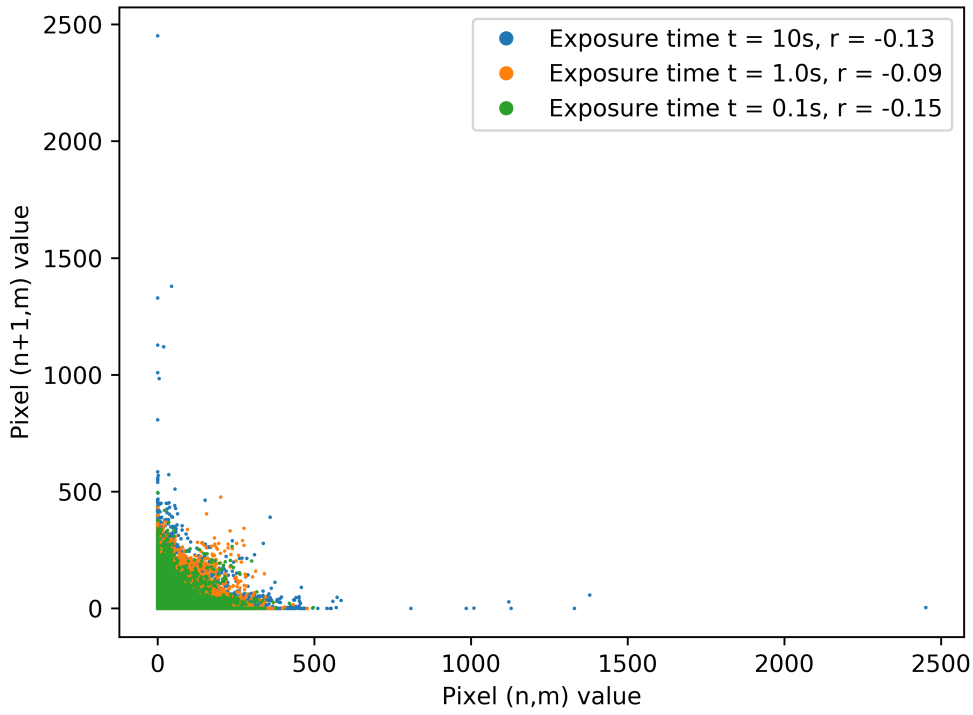


(a) Correlation before subtracting average dark frame.

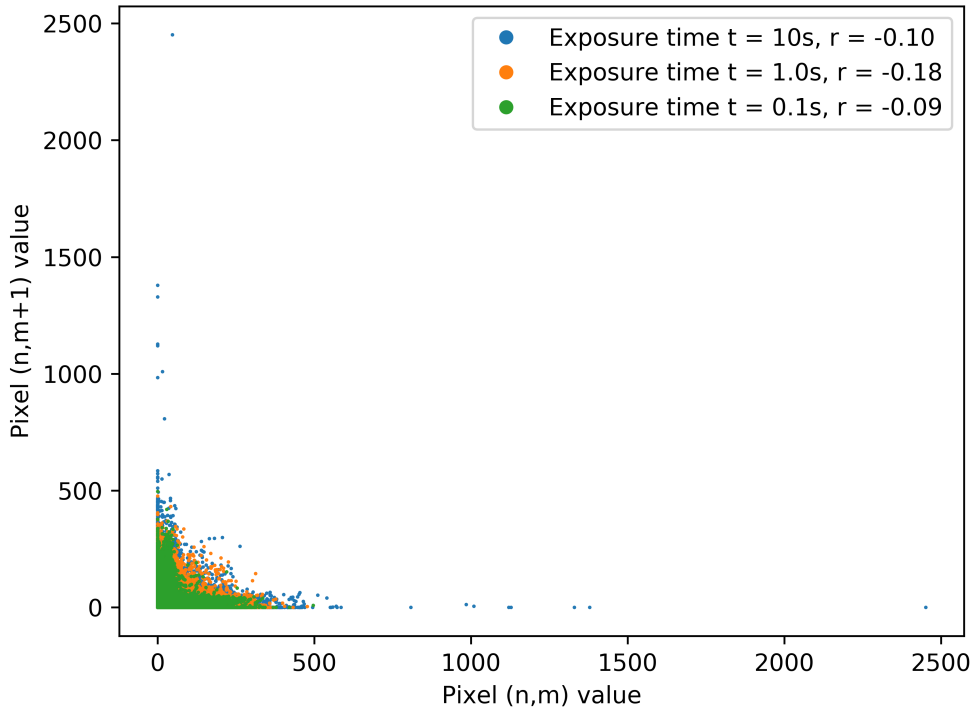


(b) Correlation before subtracting average dark frame.

Figure 3.1: Correlation between two dark frames before and after subtracting average dark frame, gain value used is 15.8.



(a) Correlation between pixels in horizontal direction.



(b) Correlation between pixels in vertical direction.

Figure 3.2: Correlation between pixels in a dark frame in horizontal and vertical directions, gain is 15.8.

star around a certain pixel by chance can be calculated as

$$P(cluster) = 1296 \cdot p^6. \tag{3.1}$$

From the definition of the normal distribution, the probability that the pixel value is within three standard deviations of the mean noise value found previously is 99.73%. The probability that it is higher than two standard deviations above the mean value is $\frac{0.27}{2}\% = 0.135\%$. Using this as the value for p in the previously derived formula 3.1, we can calculate $P(cluster)$ to be less than $7.84 \cdot 10^{-15}$. This is sufficiently low to claim that it is highly unlikely that a star can be constructed simply out of dark current noise, as each frame contains slightly over 5 million pixels. Therefore setting thresholds in the star identification code to be three standard deviations above the mean dark frame value is sufficient to eliminate the chance of this kind of noise spontaneously creating extra stars if stars are required to consist of at least six pixels. However, it must be noted that this does not protect against noise arising from all sources, such as traces left by cosmic rays.

The author of this thesis implemented both Sinclair Interplanetary's as well as Ayal's algorithms for star detection from pictures in Python. Sinclair's algorithm was used almost as described in their 2014 paper [9], with lit pixel threshold chosen as 3σ and blob size as 6. As the author did not have any excellent way to estimate a good integrated intensity threshold that would also hold well in space, it was ignored for this experiment and remains a problem that needs to be solved in future work. Ayal's algorithm was tested in two versions: one implemented the way it is described in the original thesis and another such that lit pixel threshold was lowered from 5σ above mean value to 3σ and stars were required to consist of at least 6 pixels.

3.1.5 Static Limiting Magnitude Determination Results

Using the modified Ayal's algorithm and Sinclair's algorithm, stars were detected on all test pictures. Using software developed by Astrometry.net [1], test pictures were matched to the sky. Stars in the sky that fell within the picture area were collected from Hipparcos-2 star catalogue and their theoretical on-picture coordinates were determined. This resulted in two lists of stars: one representing stars found on the picture, and the second one representing stars that actually exist in the sky.

Next, two k-d trees were constructed using Scipy's[3] built-in k-d tree implementation. One tree was constructed from the stars found on the image and the second one from stars in the sky. Using these trees, matching stars on the image to stars in the sky took place in two directions.

The algorithms described work reliably for bright stars. They also reliably miss stars that are very faint. However, a certain interval of magnitudes forms a gray area due to factors such as noise and the camera's quantum efficiency curve (the latter of which could in the future be corrected for, see Chapter 5). Within this gray area, some stars are detected by the algorithm and some are not. The k-d trees constructed by the code allowed to estimate the bounds of this gray area: the magnitude of faintest stars that were detected (optimistic limiting magnitude estimate) and magnitude of brightest stars that were not (pessimistic limiting magnitude estimate). Results are displayed in table 3.1.

[3] `https://www.scipy.org/`

Table 3.1: Limiting magnitude estimation using automated star detection

Method	Exposure time (sec)	Gain	Limiting magnitude	
			Optimistic estimate (mag)	Pessimistic estimate (mag)
Sinclair	0.1	15.8	6.3 (6.3*)	5.9 (5.9*)
Modified Ayal	0.1	15.8	6.3 (6.3*)	5.9 (5.9*)
Sinclair	1.0	15.8	8.5 (6.0*)	6.9 (4.4*)
Modified Ayal	1.0	15.8	8.5 (6.0*)	6.9 (4.4*)
Sinclair	10.0	15.8	10.3 (5.3*)	8.5 (3.5*)
Modified Ayal	10.0	15.8	10.1 (5.1*)	8.0 (3.0*)

* Equivalent magnitude for 0.1-second exposure time.

Additionally, the code written for matching was also programmed to bring out clear outliers (bright stars that were not detected, faint stars that were detected). These outliers were manually examined and boundaries of the gray area were drawn using human assistance at reasonable locations.

The code also counted the numbers of false positives and false negatives that would result from this detection routine if the following star identification code only expected stars brighter than some boundary value b. The false positive and negative values were found for b values equal to the optimistic and pessimistic limiting magnitude estimates. This means that if b is set at the optimistic estimate, then all stars brighter than magnitude b that were not detected count as false negatives, and if b is set at the pessimistic estimate, then all stars fainter than magnitude b that were detected count as false positives. False positives and negatives were also found for b equal to the arithmetic mean of the optimistic and pessimistic estimates. Results are displayed in table 3.2.

The results show that a reasonable estimate for the limiting magnitude is around 6.0, as detected by the optimistic bound on pictures with exposure time 1.0 s and supported by both bounds for exposure time 0.1 s. The very bright pessimistic bound for exposure time 1.0 s is accompanied by a very high number of false positives, meaning that a lot of the fainter stars were actually also detected, and therefore this bound should not be trusted. As for results for exposure time 10.0 s, the sheer number of stars visible on the picture likely confused the detection algorithm (especially if two stars were fairly close and happened to be linked by bright noisy pixels), as shown by the relatively high numbers of false positives and negatives for optimistic bounds and very high number of false positives for the pessimistic bound. Therefore the best estimate for limiting magnitude is still 6.0, or conservatively 5.9.

Stellar source detection was also attempted using original unmodified Ayal's algorithm. Resulting limiting magnitude for these experiments was unreasonably large with a very high false positive rate for every exposure time. Therefore, the data was discarded and the algorithm is not used in future analysis under the consideration that the algorithm in its original form is not suitable for the ESTCube-2 hardware.

Table 3.2: Numbers of false positives (FP) and false negatives (FN) for limiting magnitude estimation

Method	Exposure time (sec)	Optimistic bound		Pessimistic bound		Middle bound	
		FP	FN	FP	FN	FP	FN
Sinclair	0.1	0	7	5	4	1	6
Modified Ayal	0.1	0	7	5	4	1	6
Sinclair	1.0	4	87	123	6	62	15
Modified Ayal	1.0	5	81	125	6	64	15
Sinclair	10.0	107	153	284	5	121	58
Modified Ayal	10.0	86	199	311	7	145	81

3.1.6 Effects of the Atmosphere

One of the main differences between taking photos on Earth and taking photos in space is the effect Earth's atmosphere has on photos taken on the ground. In order to evaluate its effects, test photos were taken of six different regions of sky over a short amount of time (12 minutes). Five test photos were taken of each region. Figure 3.3 shows the locations of test images in the sky. More information about test setup can be found in Appendix A.

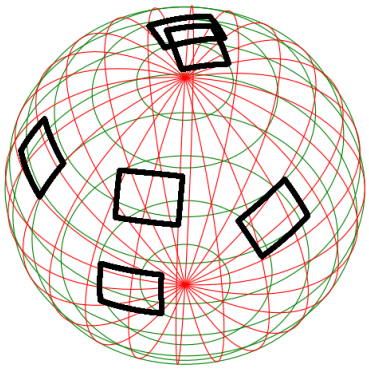


Figure 3.3: Areas of sky covered by test images, shown in the horizontal coordinate system.

Stars appear fainter to the camera when an image is taken from the ground as opposed to in orbit because some of the light is absorbed in the atmosphere. Moreover, the thickness of atmosphere is not the same when looking in all directions, but depends on how high one is looking: when a star is closer to horizon, light coming from it needs to pass through a thicker layer of air. The amount of air that light passes through is called air mass. In zenith, air mass is equal to 1. In orbit (outside of Earth's atmosphere) the air mass value would be 0. This allows for estimation of how much brighter the stars would be in orbit than on the test images obtained on the ground.

This estimation was done using Hardie's technique [32]. First, A5-type stars were extracted from the test images. For each of those stars their right ascension and declination were found. Using the location and time when the pictures were taken, zenith angle z (angle between the star and zenith) was calculated. Now air mass X was calculated for each star at the moment the picture was taken using Hardie's formula:

$$X = \sec z - 0.0018167(\sec z - 1) - 0.002875(\sec z - 1)^2 - 0.0008083(\sec z - 1)^3.$$

Next, for each unsaturated star, signal collected from the star was measured (after subtracting average dark frame) by summing up values of pixels belonging to the star. This value was converted into instrumental magnitude v using Pogson's ratio:

$$v = -2.5log_{10}(F)$$

where F is the collected signal (flux). Instrumental magnitude v is related to a star's standard magnitude V (magnitude in the visual band without the effect of the atmosphere) by the following formula:

$$V = v - k_v X + a_v(B - V) + c$$

where a_v and $B - V$ are close to 0 for A-type stars, k_v is a nightly extinction coefficient characterizing transparency of the atmosphere at the time of observations and c a zero-point constant value that is also dependent on the situation and hardware.

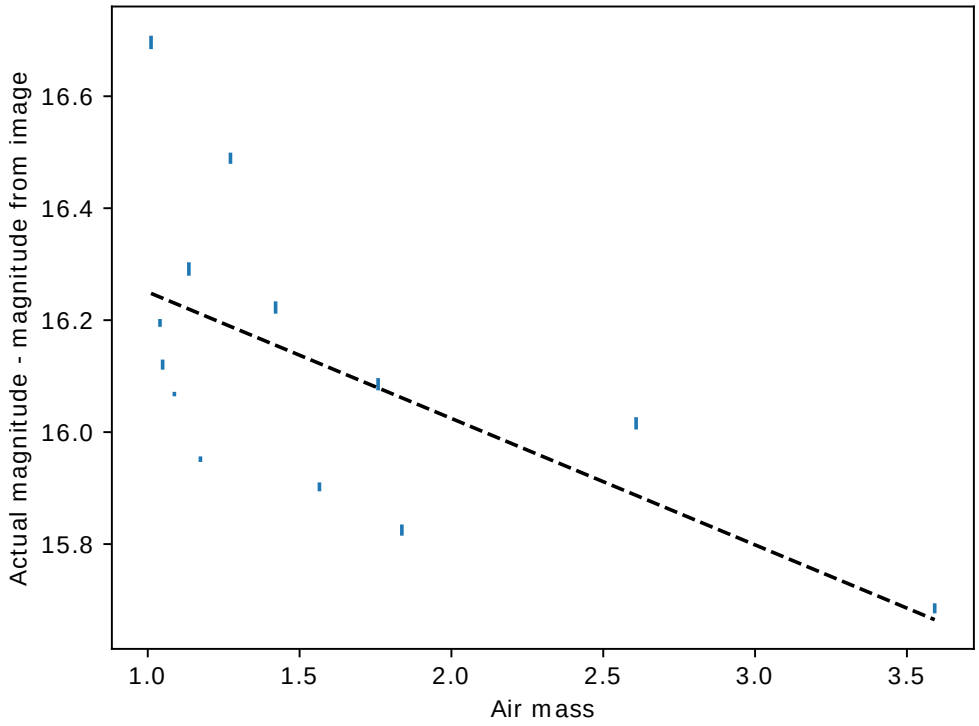


Figure 3.4: Air mass vs difference in actual and apparent magnitudes with regression line.

By approximating the value of $a_v(B - V)$ value as 0, one can calculate k_v and c values by fitting a linear function to the relation $V - v = -k_v X + c$. Standard magnitude V for every star was obtained from the SIMBAD service[4], v was observed and preprocessed, and air mass X calculated for each star using the steps described above. Figure 3.4 shows the results and also displays a linear regression line, the equation for which gives the values $k_v = 0.23$ and $c = 16.48$. Based on this relation, stars would be about 0.23 magnitudes brighter outside the atmosphere (air mass 0) than in zenith (air mass 1).

Test images in this section were taken using a gain of 8.0 instead of a maximum gain of 15.8 in order to avoid overexposing stars, which would lead to incorrect instrumental magnitude values. However, this is not a problem since the gain of the instrument only affects the value of c and

[4] `http://simbad.u-strasbg.fr/`

not the slope of the line k_v. The value of c, however, does not affect the magnitude difference between 0 and 1 air masses. Given that the test images used for calculating limiting magnitude in Section 3.1.5 were taken close to zenith (and thus with air mass approximately equal to 1), the constant found can directly be counted as a change in limiting magnitude.

3.2 Effect of Rotational Motion on Limiting Magnitude

An essential part of the ESTCube-2 mission is unwinding the electric solar sail by centrifugal deployment. This operation relies on the spacecraft rotating with speed up to 360 degrees per second [29]. When spinning, images of stars will be stretched out from fairly circular blobs to arcs. As the area over which photons from the star are spread out on the photo increases, the brightness of each individual pixel in the stellar image decreases. Although the problem of evaluating precisely how the brightnesses of all pixels will decrease is geometrically difficult, the current best estimate was obtained by using an approach according to which brightness of a star decreases approximately $\frac{l}{d}$ times, where l is the length of a long arc-shaped trail left over the course of exposure time and d the full width at half maximum (FWHM) of the blob representing the star if no rotational motion is taking place.

The formula above determines that decrease of signal level per pixel depends heavily on the size of the star image without rotational motion, which in turn depends on the magnitude of the star. Therefore in order to proceed it is first important to determine how the area of the star image depends on the star's magnitude.

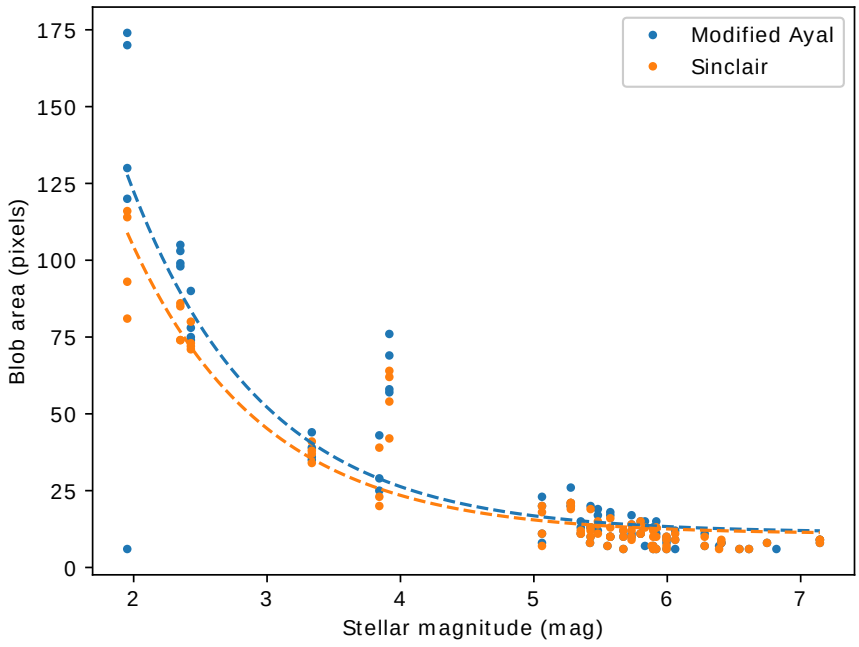


Figure 3.5: Stellar magnitude versus star blob surface area for exposure time 0.1 s.

Both modified Ayal's and Sinclair's method were used to find stars on a test image obtained using 0.1-second exposure time. During this short exposure time a star's location on the image shifts due to Earth's rotation, however, the change in location is significantly less than one

pixel and therefore the image is equivalent to a static image of the sky for the purposes of this computation. Both modified Ayal's and Sinclair's algorithm inherently calculate the blob area for each star during the process of stellar source detection. Stars were paired to their magnitudes listed in Hipparcos-2 database using the world coordinate system transformation given by Astrometry.net. This allowed to compare star blob areas to their magnitudes. Result has been shown in Figure 3.5.

The plot also displays functions fitted to the data. Per Pogson's ratio, stellar magnitude is a logarithmic function of star brightness. By approximating the relationship between stellar image area and brightness as a linear function, one can relate magnitude m to blob area A as $A = ke^{-m} + c$, where k and c are constants. Values for k and c were then determined by fitting the function using the least squares method.

Results from Section 3.1.5 show that the conservative estimate for static limiting magnitude for 0.1-second exposure time is 5.9. Section 3.1.6 results show that the limiting magnitude increases by 0.2 when pictures are taken in orbit and therefore the static limiting magnitude is $m_{static} = 6.1$. Let the rotational limiting magnitude be m_{rot}. Then m_{rot} will be $\frac{l}{d}$ times brighter than m_{static} as per the formula mentioned above (where l is the length of a star trail and d the diameter of a static image from the same star). Because m_{static} and m_{rot} are both on a logarithmic scale, then per Pogson's ratio this translates into

$$m_{rot} = m_{static} - 5 \log_{100} \frac{l}{d}.$$

Additionally, $d = \sqrt{\frac{A}{\pi}} = \sqrt{\frac{ke^{-m_{rot}}+c}{\pi}}$, where k and c were determined above.

For finding l, it is useful to remember that the satellite's fast rotation will be happening around an axis that is close to the pointing direction of the camera. Therefore $l = t\omega r + d$, where t is the exposure time (0.1 s in our case), ω is the angular velocity of the spacecraft and r is the distance of the star from the middle of the image (in pixels). The original star blob width d is also added, as $t\omega r$ only accounts for the length of the arc from the initial midpoint of the star to the final midpoint of the star. (For longer exposure times and greater ω values, the trail will form a full circle, in which case the formula no longer works.)

Putting everything together leads to the following expression:

$$m_{rot} = m_{static} - \log_{2.512}\left(\frac{t\omega r}{\sqrt{\frac{ke^{-m_{rot}}+c}{\pi}}} + 1\right).$$

By knowing the values of m_{static}, t, ω, r, k and c it is possible to solve for m_{rot} numerically. Results of this have been shown on Figures 3.6 and 3.7.

On Figure 3.6, three different angular velocities of 3, 5, 10, 20 and 40 degrees per second were observed as representative of normal, fast and extreme rotations when the satellite is not intentionally spinning fast to unwind the solar sail. The figure demonstrates how the distance of a star from the midpoint of the picture is related to whether we can see it: limiting magnitude is lower towards the edges, because the arcs formed during rotational motion are longer.

Similarly, Figure 3.7 demonstrates how limiting magnitude drops at radius 1000 pixels from the midpoint (roughly equal to half of the shorter side of the image) as the satellite starts spinning

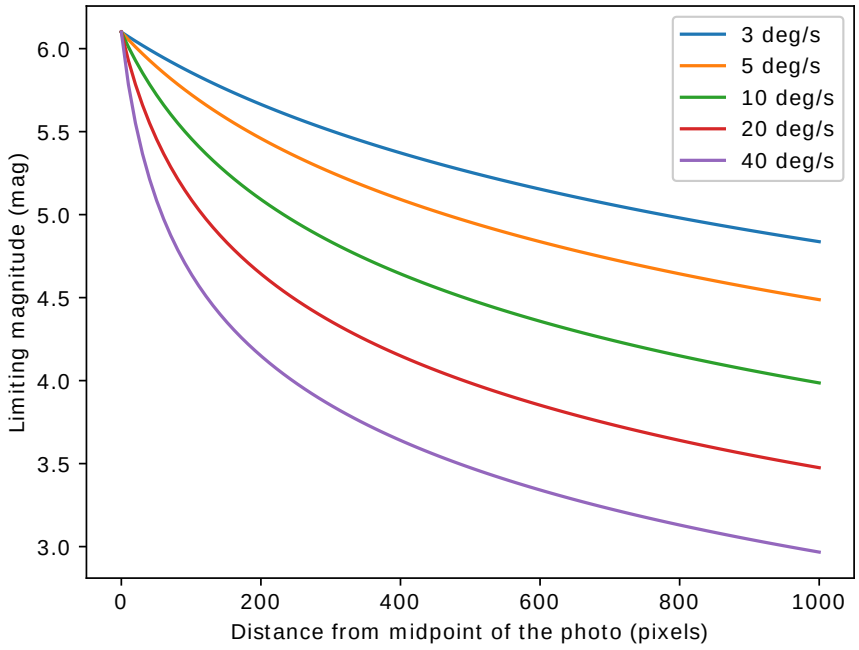


Figure 3.6: Dependence of limiting magnitude for rotational motion on the distance from the midpoint of the photo, exposure time 0.1 s.

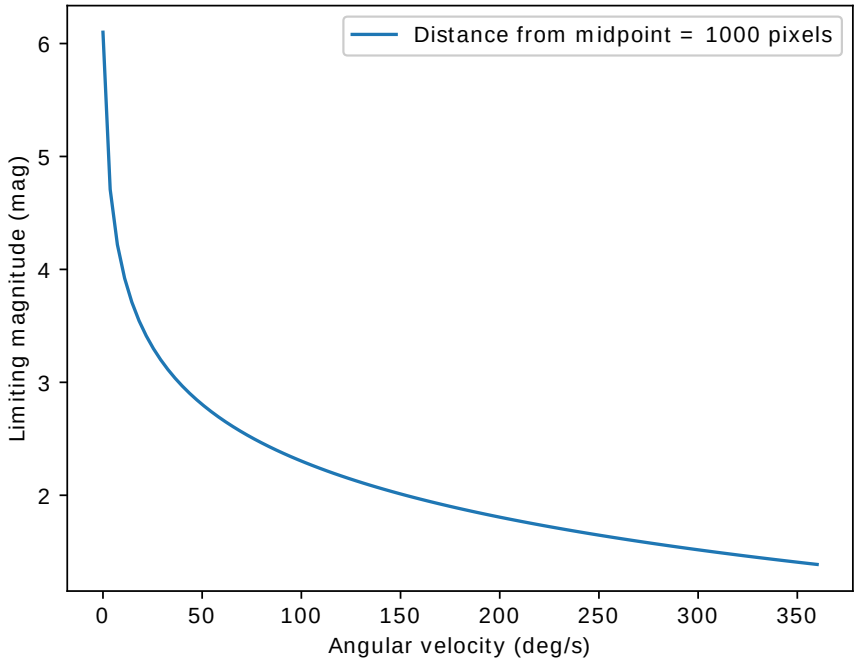


Figure 3.7: Limiting magnitude versus angular velocity, exposure time 0.1 s.

faster, up to half a rotation per second. Because of the rather steep drop, for angular velocities of 10 degrees per second or more the star tracker is only expected to be functional at occasional instances.

3.3 Star Availability

Knowing the limiting magnitude it is possible to estimate the certainty with which there are at least N stars visible in each area of the sky. Assuming the satellite is rotating at negligible speed, this was done by filtering star catalog by magnitude, simulating viewing the sky in 2 million different random directions using the star tracker camera and counting visible stars in each area of the sky. Based on results from previous sections, limiting magnitudes ranging from 4.0 to 6.0 were tested to get a good overview of star availability in both cases when the spacecraft is rotating and when it is not. Results can be seen on Figure 3.8.

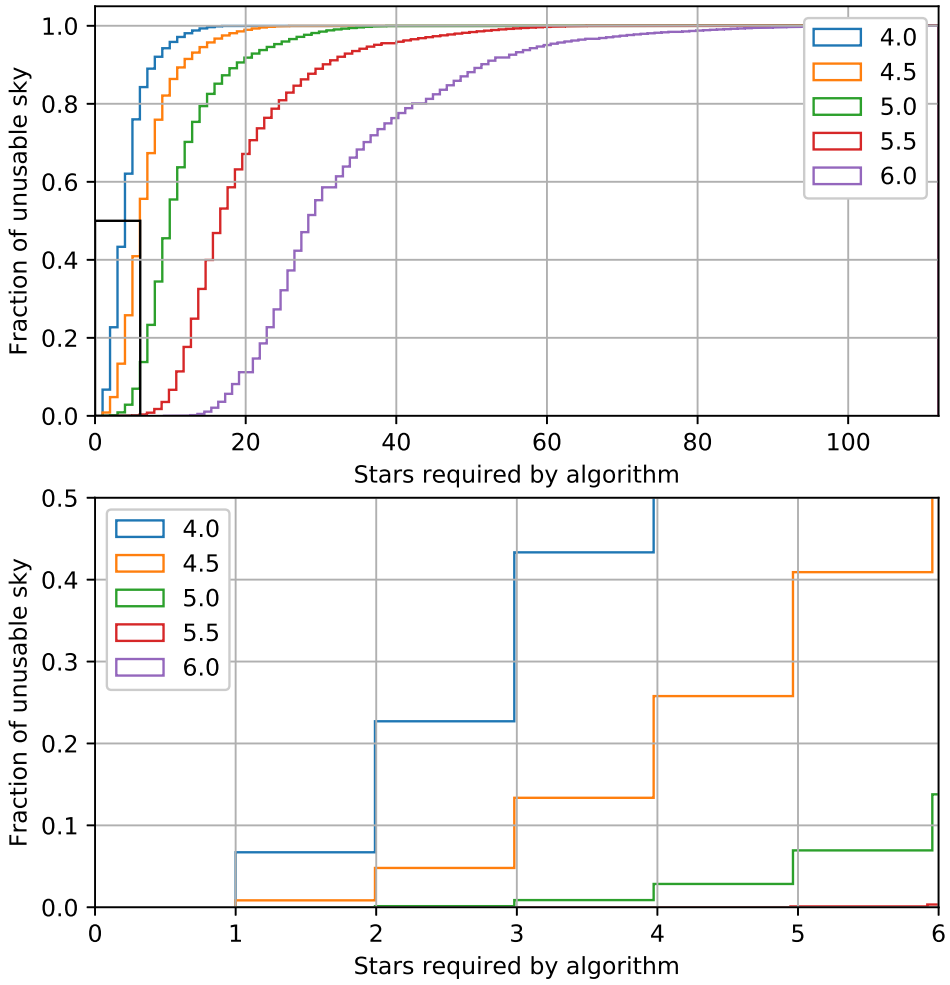


Figure 3.8: Star availability in different areas of the sky for limiting magnitudes 4.0, 4.5, 5.0, 5.5 and 6.0. Second plot represents the bottom left area on the first plot surrounded by a black rectangle.

This plot offers insight into the design choices of matching algorithm implementation. Namely,

when the spacecraft is not rotating and limiting magnitude is close to 6.0, then 10 stars or more are expected to be visible at all times under ideal conditions. However, if there is even a bit of rotational motion, then as seen on Figures 3.6 and 3.7 the limiting magnitude drops significantly, especially in the edges of the picture.

The lowest number of stars that a matching algorithm can feasibly work on is 3. However, for no rotation or slow rotation, a matching algorithm that uses four or five stars to obtain a match might perform better due to decreased chance of false positives. Based on Figure 3.8, in a situation of very slow rotation close to 100% of the sky should have at least 5 stars visible at all times, which indicates that matching based on groups of 5 stars could be a feasible solution.

When the satellite is rotating though, the situation changes significantly. For a moderate angular velocity of 5 degrees per second, magnitude of stars in the edges of the picture drops to around 4.5. Three stars of magnitude 4.5 or brighter are found in about 87% of the sky, meaning the star tracker could still work reasonably well in most cases. When the angular velocity increases to 10 degrees per second, only stars of magnitude 4.0 or brighter are reasonably guaranteed to be detected. Thus over 40% of the sky could be unusable. This leads to the conclusion that when the satellite is spinning with angular velocity greater than 10 deg/s, then the hash-based matching likely will no longer work reliably at all times. However, information provided by the star tracker can still be reliable to the attitude and orbital control subsystem, which is capable of tracking the current estimate of the position and incorporating information from other sensors to mitigate the fact that the star tracker will not be able to detect attitude on all frames.

Based on this conclusion, Chapter 4 implements matching for a static case for limiting magnitudes of 6.0, 4.5 and 4.0 and offers further insight into the feasibility and expected performance characteristics of ESTCube-2 star tracker.

4 Identifying Stars by Pattern Matching

When a star tracker takes an image of the sky, it essentially needs to solve three tasks: first, detecting stars from the image (discussed at length in Chapter 3), second, identifying which stars on the image correspond to which stars on the actual celestial sphere and last but not least, calculating the instrument's (and spacecraft's) attitude (orientation) based on the identified stars. Fortunately, thanks to observational astronomers' diligent work there exist several detailed databases of stars visible from Earth along with their celestial coordinates, which helps with solving the third task. The rest of this chapter focuses on the task of star identification.

This chapter dives into the task of star identification on the ESTCube-2 star tracker after the star detection phase is complete. It considers hash-based matching approaches and different algorithmic decisions, simulates a number of these approaches and compares the results. An important part of creating a functional algorithm is comparing the picture to a good database that has been created and organised specifically for this purpose; this chapter also analyses database creation.

4.1 Hash-Based Matching

A common approach for matching stars on an image to the database is based on geometric hashing 2.1.1. The approach has in detail been described for the case of Astrometry.net in Section 2.1.1. Although there have also been developments at using different matching schemes, the geometric hash-based ones provide a realistic way of performing the operation on nanosatellite hardware. Therefore, this thesis chooses to analyse these hash-based schemes in detail.

The hash-based approaches use groups of stars: they calculate a real-valued vector representative of the geometry of the specific group of stars. This vector is called a geometric hash. Such hashes are calculated for groups of stars found from an image taken by the star tracker camera. Then the closest match(es) to this vector is (are) found from a precomputed database, which contains hashes of some reference star groups. When a match is found from a database, the stars on the picture can be identified as specific known stars in the sky based on the database entry. An attitude estimation can then be computed for the spacecraft based on the celestial coordinates of identified stars.

One factor to consider when choosing a hash function for star identification purposes is the length of the resulting hash. This length is affected by how many stars form a group to be hashed. The length of a hash vector increases by 2 if the group size is incremented by 1. As mentioned in Section 2.1.1, group size affects the following factors:

1. larger group size generates larger hash vectors, therefore making false positives less likely;

2. larger group size means that an image taken of the sky is less likely to have a sufficient number of stars visible to the camera in the frame at once;

3. larger group size generates a much larger database, requires more memory and increases lookup time.

This thesis analyses group sizes ranging from 3 to 5.

Hashes can be normalized (as in the case of Astrometry.net, described above), meaning they are invariant under scaling, or non-normalized, meaning they contain information about the scale of the group of stars. Normalized hashes are especially good if the field of view of the camera is unknown: this way only the relative distances between stars matter and database matches can be found even without knowing how many pixels correspond to what distance in the sky. However, in the case of ESTCube-2, the camera's field of view is in fact known and will not change. Using non-normalized hashes reduces the chance of false positives by essentially adding an extra filter. At the same time it does not increase database size, unlike increasing group size, which also requires additional stars to be in the field of view. Therefore, in the case of ESTCube-2 a non-normalized version is likely to perform better.

Hashes could also contain information about star brightness or treat all visible stars the same, ignoring their magnitude. There are several ways to approach this. One way would be to store the magnitude of every star in the hash and compare the magnitudes obtained from the image to the magnitudes in the hash. Another way would be to rank stars in the hash by brightness and store this information: this has the advantage of not increasing the database size as much (should storage space become a problem), however, it requires a good solution for the cases where star magnitudes are fairly close to each other. Multiple other ways, such as storing the brightnesses of brightest and/or faintest stars in the group only, also exist.

All of the listed methods of taking brightness information into account, however, require a good characterization of the quantum efficiency of the star tracker camera, an adjustment of the magnitudes from a standard database such that they would reflect how bright the camera sees them and an estimate of the plausible error rate. As such, it falls outside the domain of this thesis. However, as this type of characterization and database adjustment is necessary in any case and should happen before ESTCube-2 flight, this method of modifying hashes and improving matching performance can be analysed in future work.

4.2 Database Generation

Hash-based approaches rely on comparing hashes of groups of stars extracted from the picture with hashes of star groups calculated based on their known locations. This section considers the generation of a database of such hashes calculated based on known locations.

4.2.1 Experiments in Increasing Generation Speed

The easiest way of generating a database of star group hashes for matching purposes is to take a list of all stars in the sky that are bright enough to be visible to the star tracker camera (call this set S), form all possible groups of stars that have N stars in it (where N is 3, 4 or 5, depending on the kinds of hashes one wishes to use) and check for every group whether it fits within the star tracker's field of view. In fact, this method was used in the first star tracker simulations conducted before the involvement of the author of the current thesis. However, this approach is unwise due to the fact that this algorithm requires checking $|S|^N$ groups of stars, where $|S|$ is the size of the catalog. If the limiting magnitude is chosen as 4.5, then the catalog size is 837, which already takes an unwisely long time to generate for the entire sky. With limiting magnitude increased to 5.5, the catalog size $|S|$ increases to 2617 and therefore algorithm runtime also increases 30 times for groups of size 3 and 300 times for group size 5.

A slightly optimized method developed by the author utilizes the possibility of discretising the sky into spherical surface compatible pixels, built-in into Python's Astropy and Healpy libraries. A lookup table was created that maps a pixel number to an array of bright stars located inside the pixel. In order to form the groups of stars, each pixel was examined together with its neighbouring pixels (to cover an area with diameter greater than or equal to the diameter of field of view). Using stars from these particular pixels, all possible groups of appropriate size were formed. Each group was then tested for whether or not it fit into the star tracker's field of view.

The described method has the same time complexity as the previous method and achieves the same results, however, the constant factor for the two methods differs significantly. With a field of view as small as the one ESTCube-2 star tracker camera has, this method speeds up database generation and can be used to run the analysis at a significantly improved pace.

4.2.2 Choosing Reference Groups

Another choice to be made is whether or not to store hashes of all groups of chosen size that fit inside the field of view in the database or to choose a subset of them. Naively storing all possible hashes in the database has the advantage that every hash calculated from the image must have a match in the database (assuming stars have been detected correctly in the first place). However, in practice it is unwise for several reasons.

First, storing hashes of all groups of stars that are brighter than the limiting magnitude and fit inside the camera's field of view means that the corresponding database becomes huge. This is especially true for larger group sizes and should already disqualify the idea of storing all possible hashes in the database. Even more concerning is the fact that bright stars are not distributed uniformly across the sky. If hashes of all groups of appropriate size are stored in the database, then the areas of sky with more stars in them are mathematically more likely to get matches, including false positives. This means that the output of the matching routine becomes biased towards areas with lots of bright stars. Therefore it is essential to make some selection regarding the hashes of which groups will be stored in the database.

Not storing all the hashes creates a problem where if one just chooses a random group of appropriate number of stars from the image, this group's hash might not have been incluced in the database at all. This could increase both false negatives (the group does not find a match at all) as well as false positives (group from the image is matched to a wrong one from the database that has a similar enough hash). This problem can be solved by calculating hashes of several groups of stars from the picture, calculating attitude estimates based on each of them and requiring that several of them give the same attitude solution.

The implementation created as part of this thesis uses the following method of creating a reference database. First, a number of reference stars were selected from the database using a Monte Carlo method. The reference stars were chosen such that at least $N = 6$ stars would be visible regardless of the pointing direction of the camera. As demonstrated on Figure 3.8, in nearly all of the sky this means stars of magnitude 5.5 or brighter. These stars comprise a list of candidate stars for reference group formation.

Next, a modified version of the faster full database generation method was used. For every Healpix pixel p in the sky (with Healpix $NSIDE$ parameter chosen as 32), this pixel and its neighbouring pixels up to a certain distance were examined. Stars belonging to those pixels were collected and ordered by magnitude. Then, groups of stars were generated such that star groups consisting of brightest stars were generated first and fainter stars were added incrementally. For every such generated group, the code checked whether the midpoint of its diameter (defined as the line between the two stars furthest apart) fell within pixel p. If so, it was added to the database. For every pixel, N such groups were identified (with $N = 3$ for this specific experiment). When N groups satisfying the conditions were found, further group generation for this pixel p was aborted and the code moved on to repeating the process for another pixel.

As a result of this database generation method, it was assured that reference hashes in the database represented groups that were roughly evenly distributed over the celestial sphere. Three different databases were created for group sizes of 3, 4 and 5. These databases were then used in the matching process.

4.2.3 Database Structure

The database of hashes can be rather large and lookups into this database happen constantly. Thus, speeding up the common operation of hash lookup from the database would increase performance significantly. To achieve this, it is useful to organise entries in the database by regions of sky (for example, by which Healpix pixel their midpoint belongs to). In this case if the star tracker is given information about the spacecraft's approximate attitude, then it can determine which pixels of the sky would be the most likely candidates for containing a match and determine which lines of the database correspond to these pixels.

Moreover, it is actually possible to obtain approximate attitude information from the Attitude and Orbital Control Subsystem (AOCS). This estimate can then be used to index into the correct region of the database in the described manner. If a sufficient match is not found, surrounding regions can be searched, incrementally moving further from the initial estimate. This search can be broadened until a certain time allocated for the computation is exceeded and then terminated.

This system needs to perform a computation mapping the attitude estimation into a Healpix pixel, an operation that may not be trivial on the star tracker FPGA. However, since it is an operation that only needs to be performed once every time a picture is obtained, it can be handled by the star tracker CPU or even AOCS without having too large of a cost.

4.3 Matching Images to Database

In orbit, the star tracker needs to take stars detected on an image, form groups of them, calculate hashes and compare these hashes with database entries. An image may contain a lot of visible stars and therefore it may not be wise to form groups of all of them. Instead, star brightnesses become useful: as stellar image sizes in pixels correlate to their magnitudes, the star tracker can find some certain number of the brightest stars from the image and calculate hashes based solely on those stars. It is also possible to iteratively calculate an increasing number of hashes and look up their closest matches from the database until a certain threshold in terms of computation time has been reached, after which the lookup may be terminated and the procedure may return the best attitude estimate it has (or announce a lack thereof). If used together with the previously described procedure of broadening the areas of database in which lookups are performed, then these two incremental procedures may also be interleaved.

The author of this thesis inherited a naive Python implementation of a matching procedure from previous collaborators working to improve ESTCube-2 star tracker. This naive implementation followed six steps:

1. gather all stars detected on an image using astrometry.net that are brighter than magnitude 6.0;
2. read in a previously generated database;
3. combine stars found from the image into all possible groups of 4;
4. calculate a normalized hash for each group;
5. for each group, find the first entry from the database where each real value in the hash vector does not differ by more than a constant from the hash value calculated from the image (this constant threshold value determined experimentally based on a very small sample size to get exactly one correct match);
6. compare all matched stars to real stars known to be found on the picture (identified using astrometry.net) and report if any of the matched stars actually exist on the picture.

This procedure was improved on by the author of the current thesis in steps 3-5.

First of all, instead of generating all groups of 4 stars from the picture and then moving on to matching phase, stars were sorted by brightness and groups were generated starting from brightest stars, as described above. Group size requirement was also modified such that it could accept groups of sizes 3, 4 or 5 and deal with normalized as well as non-normalized hashes.

Next, the matching to the database phase was moved such that matching each group would take place right after the group is generated and before next ones are generated. The code was also modified such that the group is compared to the whole database for finding the best match, instead of accepting the first one that falls under a constant threshold. The requirement that hash values may not differ by more than the constant threshold value was left in place to ensure that in cases where there is no match in the database no false positives are created. However, different threshold values were experimented with.

Other steps also required minor modifications due to changes in hash length in experiments with different group sizes and/or non-normalized hashes.

Lastly, a timer was added to the code. This timer measured time spent on generating groups and finding matches from the database. When a fixed amount of time had passed, the procedure was terminated and results returned.

While the process could have been sped up by indexing into correct regions of the database corresponding to approximate attitude estimate, this was not implemented as the increased performance would not have given enough meaningful information. Namely, this Python simulation running on a laptop is different enough from the system on which this matching phase will actually be taking place. As such, the time allowed for the matching procedure can also not be directly translated into time that it will be given on the star tracker. Instead, timer values that would give a good overview of the performance characteristics were used.

Parameters used in the simulation and results obtained are discussed in Section 4.4.

4.4 Matching Performance

In order to determine the expected performance of slightly different versions of the matching algorithm, simulations were run on test images with various different parameters.

Matches were identified for all possible combinations of the following parameters:

- six different test pictures from different areas of the sky, each with exposure time 1.0 s and gain 8.0;
- three different maximum allowed matching times of 10.0 s, 1.0 s and 0.1 s;
- ten different hash difference threshold values (by which each value in the hash vector was allowed to differ from the real value);
- three different group sizes of 3, 4 and 5;
- normalized and non-normalized versions of hashes;
- three different limiting magnitude values of 6.0, 4.5 and 4.0.

For each test, the number of true positive (TP) and false positive (FP) matches was found and precision was calculated based on those counts as $\frac{|TP|}{|TP|+|FP|}$.

Additionally, the same tests were run on a picture taken of the same region as one of the six aforementioned pictures, but with exposure time of 0.1 seconds and gain of 15.8. Comparing the results for this picture to the results for the other picture displaying the same area (but obtained with different exposure time and gain values), showed that there was no significant difference. This is due to the fact that the main difference, the number of visible stars, is compensated for by limiting magnitudes of stars used in the matching database. Therefore the results obtained for six test images used are still applicable for ESTCube-2 in space, where exposure time and gain values will be 0.1 s and 15.8, respectively.

The ten hash difference threshold values used were calculated such that the results between normalized and non-normalized implementations would be comparable. For this, the normalized threshold values (where the distance between two stars furthest apart was normalized to $\frac{\sqrt{2}}{2}$) were chosen as constants c ranging from 0.0025 to 0.3. For non-normalized versions, threshold value for each group was chosen to be

$$\frac{\sqrt{2}}{2} \cdot c \cdot d$$

where d is the distance between two stars furthest apart in the group and c the same constant as used in the normalized case.

Sections 4.4.2, 4.4.3 and 4.4.4 discuss the simulation results.

4.4.1 Normalized vs Non-Normalized Cases

As expected, the number of correct matches for normalized and non-normalized cases was roughly equal in all cases. However, the number of false positives was significantly less for all non-normalized cases. Therefore it would be wise to use non-normalized hashes in the final version of ESTCube-2 star tracker software.

4.4.2 Maximum Allowed Matching Times

Maximum allowed matching times tested were 10.0 s, 1.0 s and 0.1 s. After the allotted time had passed, the matching procedure was terminated, as described in Section 4.3. As an example, results for one specific test picture for limiting magnitude of 4.0, hash difference threshold value of 0.03 and non-normalized hashes have been given on Figure 4.1.

The example results shown indicate that by letting the matching procedure run for a longer time, matching precision decreases significantly. This is likely due to two factors: matching procedure starts by forming groups of the brightest stars on the picture, and database creation favours bright stars. Thus, the groups matched in the beginning of the procedure are likely true positives. As the matching procedure progresses, however, the groups found from the picture no longer exist in the database. This is because the database (in this case) is limited to stars of

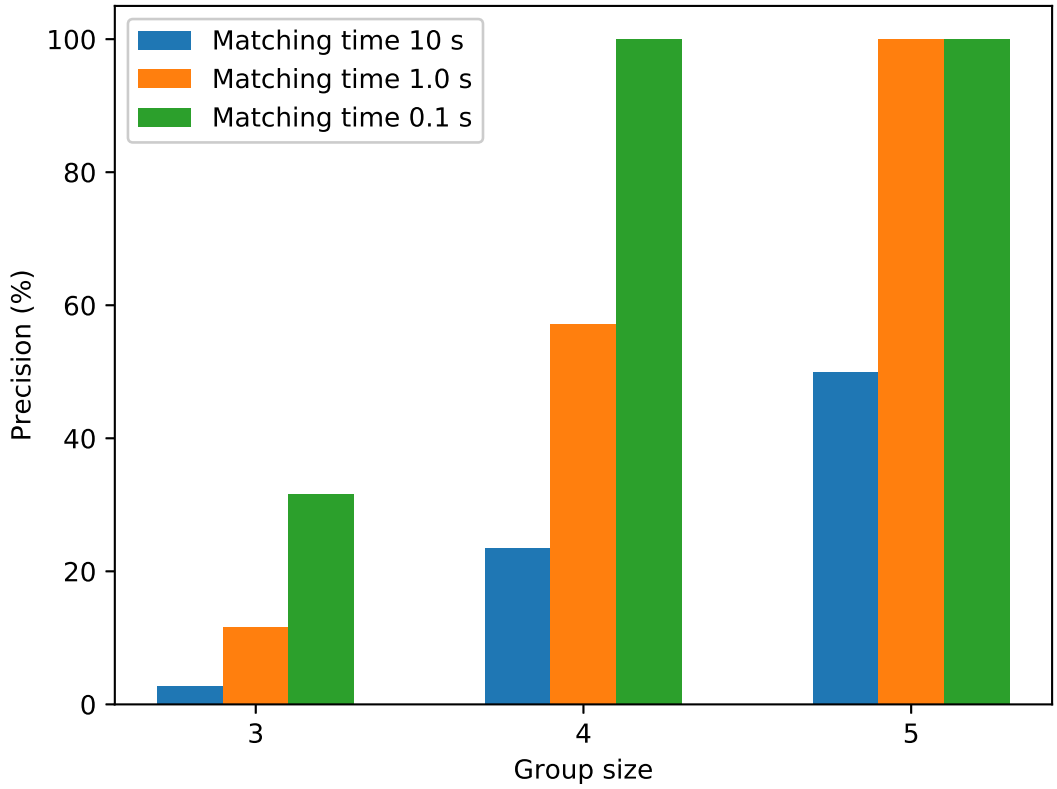


Figure 4.1: Maximum matching time vs precision for all group sizes, for non-normalized hashes, limiting magnitude 4.0, hash difference threshold 0.03, on a picture taken towards Ursa Major α and β.

magnitude 4.0 or brighter, and even when it is not, the number of groups entered to the database per sky area is constrained. Therefore when bright stars exist in the area, database entries from that area also use these bright stars.

It is important to note that different pictures, limiting magnitudes and hash difference thresholds also exhibited similar behaviour. Thus, allowing the matching procedure to run for a longer time gives no extra advantage.

4.4.3 Hash Difference Threshold Values

The hash difference threshold value indicates by how much each value in the hash vector of a group of stars from a picture may differ from hash vector values in the database. If the difference for each hash vector value is smaller than the threshold value, then the two hashes may be considered to match (unless there is a better match with the database).

This reduces false positives by allowing the star tracker to return a “no match” result when no sufficiently good match is found. It also allows the star tracker to match several different groups

of stars from the picture to achieve a better estimate, even when some of these groups do not have corresponding database entries at all. Thus the groups that do not have database matches will not affect the estimate determined by other (true) matches from the same picture.

Figure 4.2 shows an example of how this threshold value affects precision. While threshold values tested ranged from 0.0025 to 0.3, results are only displayed for threshold values up to 0.1, as the results for greater values were as bad or worse than the result for threshold 0.1. The missing beginnings of lines on the figure indicate areas where no matches were detected and precision could therefore not be calculated.

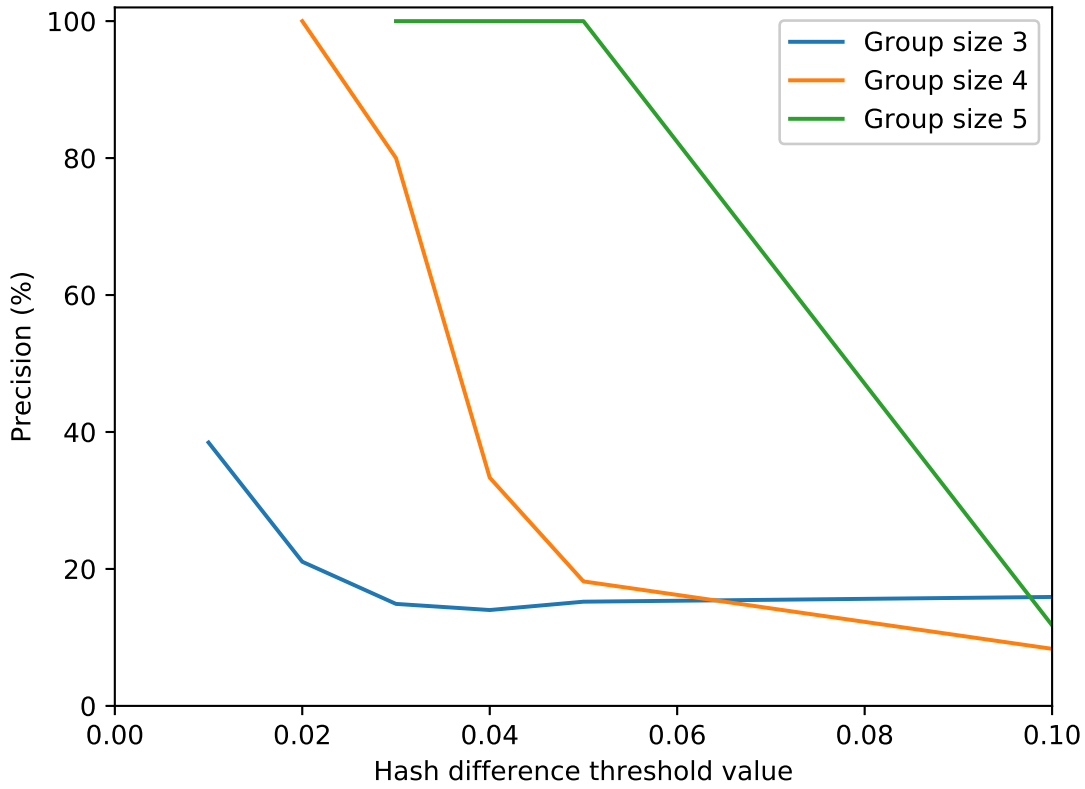


Figure 4.2: Relationship between hash difference threshold value and matching precision. Limiting magnitude 6.0 mag, matching time 1.0 s, on a picture taken towards Ursa Major α and β.

Results show that the hash difference threshold value is especially relevant for smaller group sizes. While the threshold value of 0.05 may be sufficient for groups of size 5, for groups of size 3 the threshold needs to be tighter (for example 0.01). Results for other combinations of input parameters also showed similar behaviour for the threshold value.

4.4.4 Limiting Magnitude Values

Limiting magnitudes ranging from 4.0 to 6.0 severely affected database size for all group sizes. This can be seen on Figure 4.3.

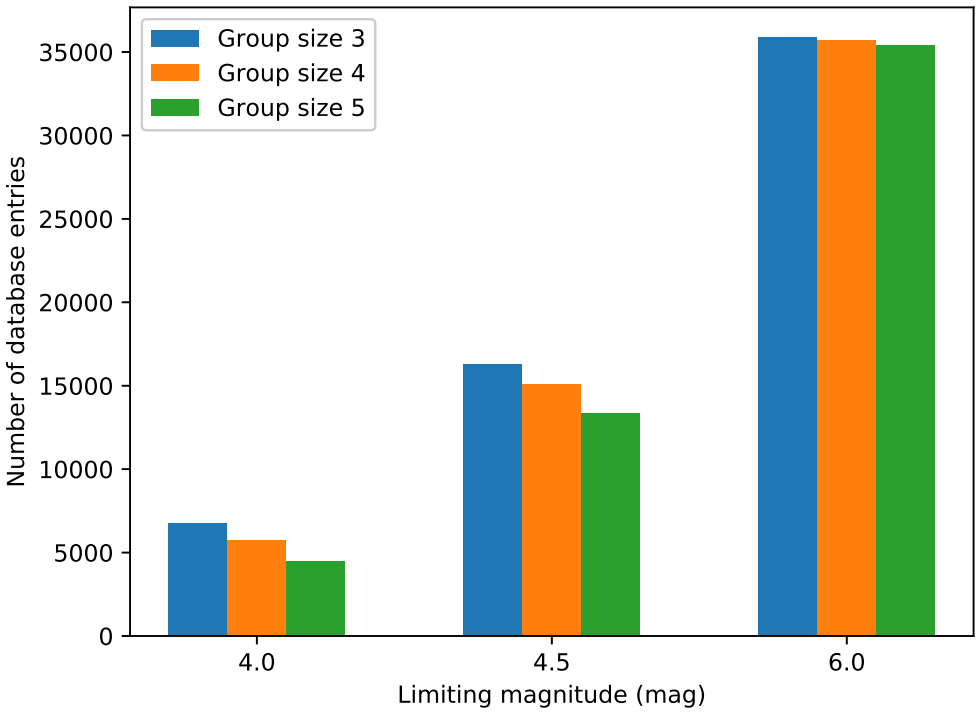


Figure 4.3: Number of database entries in the matching database for different group sizes and limiting magnitudes.

Table 4.1 shows matching precision for six different test pictures of different sky areas for group sizes 3 and 4, matching time of 0.1 s, non-normalized hashes and hash difference threshold values of 0.01 and 0.03. The existence of any positive matches has been indicated with green. As can be seen from the table, setting a rather small hash difference threshold value increases the precision of group size 3 hashes, but may become too limiting for group size 4. A more relaxed threshold value, however, leads to lower precision.

The precisions listed in the table show matching performance in cases where no *a priori* attitude estimate is known. However, as the star tracker will be operating together with the rest of the attitude and orbital control subsystem, it does not have to solve the lost-in-space problem in full. This is expected to increase matching precision when used on the spacecraft.

Results also demonstrate that limiting magnitudes in the range of 6.0 to 4.5 were entirely sufficient to obtain matches in the test pictures. Limiting magnitude dropping to 4.0 only hindered the performance somewhat for these particular test pictures and is therefore a promising sign for the phase of ESTCube-2 mission when the spacecraft is rotating rapidly. However, it must be taken with a word of caution: the sample comprising these six test images may not be representative of the whole sky.

Table 4.1: Matching precision for six test pictures of different sky areas, group sizes 3 and 4, non-normalized hashes, matching time 0.1 s, hash difference threshold values 0.01 and 0.03, existence of any positive matches indicated with green

Hash difference threshold value 0.01

Picture	Group size 3 Limiting magnitude (mag)			Picture	Group size 4 Limiting magnitude (mag)		
	6.0	4.5	4.0		6.0	4.5	4.0
0	100%	80%	66.7%	0	No matches	No matches	No matches
1	66.7%	85.7%	100%	1	100%	100%	100%
2	100%	50%	No matches	2	No matches	No matches	No matches
3	100%	100%	100%	3	100%	100%	100%
4	50%	50%	100%	4	No matches	No matches	No matches
5	80%	85.7%	100%	5	100%	100%	No matches

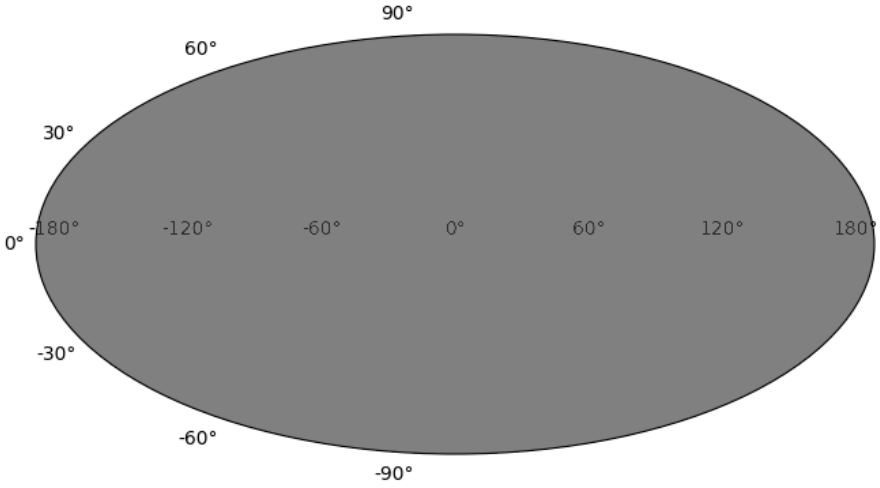
Hash difference threshold value 0.03

Picture	Group size 3 Limiting magnitude (mag)			Picture	Group size 4 Limiting magnitude (mag)		
	6.0	4.5	4.0		6.0	4.5	4.0
0	60%	41.7%	31.6%	0	66.7%	75%	100%
1	50%	80%	72.2%	1	100%	100%	100%
2	33.3%	22.2%	0%	2	No matches	No matches	No matches
3	100%	80%	53.8%	3	100%	100%	100%
4	20%	9.1%	5.3%	4	No matches	No matches	No matches
5	66.7%	45.5%	20%	5	100%	90%	50%

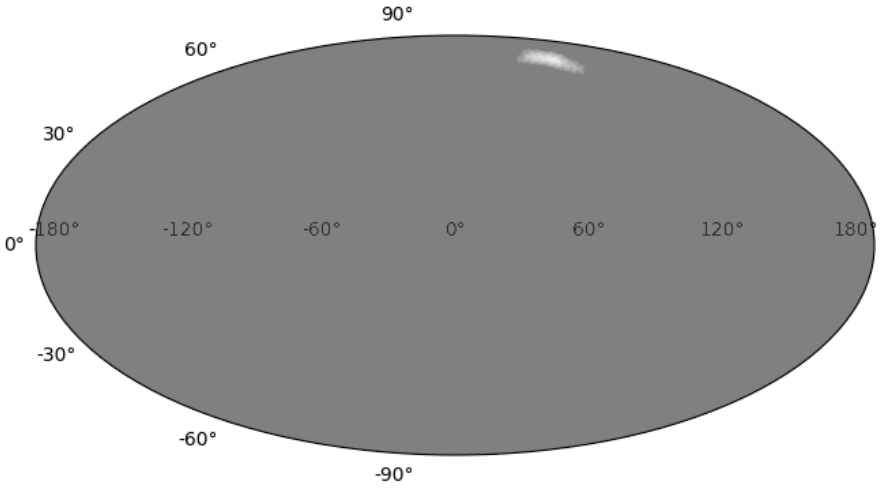
In order to be able to tell which sky regions would be problematic for the star tracker, code behaving similarly to the one used in Section 3.3 was used to simulate the star tracker pointing its camera at randomly chosen regions of the sky and detecting whether any of the groups from the generated database fall within its field of view. Results for group size 3 and limiting magnitudes of 6.0, 4.5 and 3.0 have been shown on Figure 4.4, which shows the areas for which at least one reference group was found from the database in gray.

These maps demonstrate that while the star tracker will lose its ability to determine attitude from every single frame as limiting magnitude drops, it remains functional on certain areas of the sky even at relatively low limiting magnitudes. For interpreting these maps it is important to realize that there might be occasional frames with an insufficient number of bright stars within the gray areas of found matches. These frames are overshadowed by the fact that most of the frames in nearby directions did achieve good matches. Thus, they are more useful for pointing out problematic areas of the sky for certain limiting magnitudes. With that said, the maps do display that occasional matches are achieved in various areas of the sky even for limiting magnitudes as low as 3.0.

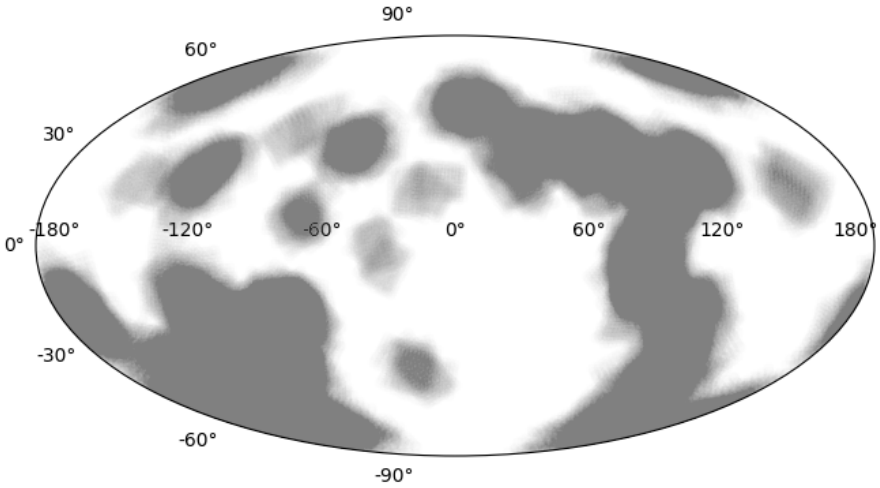
As demonstrated on Figure 3.6, magnitude 3.0 corresponds to the faintest stars visible near the edges of the picture when the satellite is rotating with angular velocity 40 degrees per second. The maps on Figure 4.4 show that the star tracker can still provide useful information to the rest of the system part-time, even under these extreme circumstances.



(a) Limiting magnitude 6.0.



(b) Limiting magnitude 4.5.



(c) Limiting magnitude 3.0.

Figure 4.4: Areas of the sky that are capable of giving a match for group size 3 with various limiting magnitudes. Match-capable areas shown in gray.

5 Discussion and Future Work

This thesis demonstrates that the ESTCube-2 star tracker can reasonably be expected to detect stars up to magnitude 6.0 when the spacecraft has a small angular velocity. This limiting magnitude drops significantly as the spacecraft starts spinning rapidly for the purpose of the electric solar sail deployment. The change in limiting magnitude is especially significant in the edges of the camera frame, where the limiting magnitude drops close to 4.5 for the angular velocity of 5 deg/s, to 3.5 for angular velocity of 20 deg/s and below 2.0 for 360 deg/s, as seen from Figure 3.7. Based on these limiting magnitude estimates, the star tracker is expected to achieve an accurate attitude estimate at least 87% of the time for angular velocities up to 5 deg/s (as demonstrated by Figures 3.8 and 3.6).

Matching simulation results show that obtaining a precise match for limiting magnitudes 6.0 to 4.5 is best done using non-normalized hashes of 3 or 4 stars, starting the matching procedure from the brightest stars found on the picture and significantly limiting the number of groups of stars from the picture that are compared to the database. In the case of the experiment conducted and for the specific example database generated in this thesis, this meant not letting the matching procedure run for longer than 0.1 s, but this limit needs to be determined in the future specifically for the star tracker hardware and finalised database.

The number of stars in a group to be matched with the database might depend on the specific mission phase: for small angular velocities, the group size of 4 might be better, but for faster ones the size needs to be reduced to 3. Therefore it might be good to store databases for both options on the satellite and switch between them as necessary.

A lot of work still needs to be done before ESTCube-2 can fly with a functional star tracker. This thesis offers guidelines for doing so from the algorithmic perspective. However, there is also a number of questions that still need answers before the star tracker is ready for space.

This thesis largely focuses on determining the limiting magnitude and its implications. Stellar magnitudes, however, depend on the instrument by which they are being observed. This can be adjusted for by characterizing the camera's quantum efficiency and re-calculating the stellar magnitudes from catalogues based on the stars' colour indices.

The performance of a star tracker also largely depends on how well various optical distortions can be corrected for during the flight. Functions correcting the locations of stars detected on the picture can be created before the flight based on the characterization of the optical system's geometric distortions. Additionally, issues arising from chromatic aberration [33] need to be analysed further.

The star tracker camera uses an electronic rolling shutter to control pixel exposure. This causes issues in cases of rapid rotation. The effect of a rolling shutter can be corrected for in cases of known angular velocities [9] and should be implemented for the ESTCube-2 star tracker.

Code in this thesis has all been programmed in the Python programming language. While some tasks like database generation are not conducted on the satellite and can therefore remain in Python or any other language, code that needs to run directly on the star tracker needs to be synthesized for the FPGA or rewritten in C++ for the CPU. There has been some previous work regarding the FPGA implementation [30], but the FPGA code that has already been written was not able to be synthesized. Additionally, work done in this thesis shows that there may be a need to modify the code that has already been written for better performance.

While some hashing schemes have been covered thoroughly in this thesis, there are still areas that should be explored further. Section 4.1 points out the advantages of including brightness information in the hashes. These options should be explored further in the interest of achieving higher matching precision.

Another complicated issue is star image centroid detection in cases of long arc-shaped trails caused by rapid rotation. Especially on an FPGA, this presents a challenge that needs to be solved in order to be able to have a functional star tracker for great angular velocities.

Section 4.2.3 describes a way of speeding up the matching process by organizing the database of hashes by location and creating a mapping from celestial coordinates into the corresponding database region. The precise implementation of this needs to take place after the hash database contents have been finalised.

Finally, while this thesis focuses on detecting stars simply based on star signal being above the signal-to-noise ratio, it might be possible to explore an alternative method of achieving a higher limiting magnitude for fast rotation. Knowing the precise angular velocities of the satellite in all directions, it might be possible to use some convolution matrix to get rid of the effects of the rotation and gather signals from the stars even if the pixels illuminated by the star are indistinguishable from the noise by themselves. However, this hypothesis may prove to be unwise, as simply reducing the signal-to-noise ratio might be easier.

6 Conclusion

This thesis focused on the feasibility of the ESTCube-2 star tracker and explored its limitations in cases of a rapidly rotating spacecraft. It explored the functionality and implementation details of already existing and tested nanosatellite star trackers and brought out the functional requirements for the ESTCube-2 star tracker. Based on the obtained information it then examined the performance of the star tracker for extreme as well as moderate and mild angular velocities.

The thesis determined the brightness of the faintest stars that can be detected by the star tracker (called the limiting magnitude) by taking test pictures of the sky, implementing two star detection algorithms and running them on the test pictures. It also provided a way of measuring the number of false positives and false negatives for star detection. Although the test pictures were taken on Earth and star brightnesses were thus decreased due to extinction in the atmosphere, this was corrected for using well-established astronomical methods. Next, the thesis derived a formula for calculating how much the limiting magnitude changes due to satellite's rotation and applied it to the static limiting magnitude determined experimentally from the test pictures.

Limiting magnitude was used for evaluating algorithmic choices in the star identification process. Stars detected on the images taken by the star tracker were matched to a database of actual star patterns to obtain the celestial coordinates of stars on the picture. When used on the spacecraft, these coordinates can then be used to identify the spacecraft's attitude. The thesis explores the hash-based matching scheme that has become an industry standard and identifies a range of parameter choices that are likely to give good performance for ESTCube-2. It also analyses the increased performance that can be achieved by creating a good matching database, describes how to do so and generates several distinct ones for use with different parameters. Last but not least, the thesis improves on an existing naive matching implementation, incorporates options to set a significantly larger number of parameters and uses it to evaluate matching performance on a number of test pictures.

The results show that using optimal parameters established with the help of simulations, the star tracker is capable of continuously determining the spacecraft's attitude with a frequency of 10 Hz for angular velocities less than 5 degrees per second around the roll axis more than 87% of the time. Moreover, since the spacecraft does not rely solely on the star tracker for attitude determination, it is possible that the star tracker can provide valuable information to the attitude and orbital control subsystem in the case of greater angular velocities as well. This is a better estimate than the allowed angular velocities for several state-of-the-art commercial standalone nanosatellite star trackers that cannot rely on information from other sensors simultaneously. Therefore the custom-built star tracker is likely to be a valuable resource for the spacecraft for attitude determination purposes.

Acknowledgements

I would like to thank

supervisors Tõnis Eenmäe and Hendrik Ehrpais for their guidance and support,

the ESTCube-2 team for their enthusiasm and kindness,

Tuuli, Heiki and my mom for providing me with food while I was buried elbows deep in the code,

Janno for suffering with me,

and Sander and Phil for their excellent constructive criticism.

Bibliography

[1] D. Lang, D. W. Hogg, K. Mierle, M. Blanton and S. Roweis. Astrometry.net: Blind astrometric calibration of arbitrary astronomical images. *The Astronomical Journal*, 139:1782–1800, May 2010.

[2] H. Ehrpais, I. Sünter, E. Ilbis, J. Dalbins, I. Iakubivskyi, E. Kulu, I. Ploom, P. Janhunen, J. Kuusk, J. Šate, R. Trops, and A. Slavinskis. Estcube-2 mission and satellite design. *Small Satellites Systems and Services Symposium*, 2016.

[3] I. Iakubivskyi, H. Ehrpais, J. Dalbins, E. Oro, E. Kulu, J. Kütt, P. Janhunen, A. Slavinskis, E. Ilbis, I. Ploom, I. Sünter, R. Trops, and M. Merisalu. Estcube-2 mission analysis: plasma brake experiment for deorbiting. *67th International Astronautical Congress*, Sept. 2016.

[4] S. Beck. Star camera. *Space Technology 6, New Millennium Program*. `https://web.archive.org/web/20110721054014/http://nmp.nasa.gov/st6/TECHNOLOGY/star_camera.html`. 2019-05-18.

[5] C. C. Liebe. Star trackers for attitude determination. *IEEE Aerospace and Electronic Systems Magazine*, 10(6):10–16, June 1995.

[6] C. C. Liebe. Pattern recognition of star constellations for spacecraft applications. *IEEE Aerospace and Electronic Systems Magazine*, 8(1):31–39, Jan 1993.

[7] J. L. Junkins, C. C. White, and J. D. Turner. Star pattern recognition for real time attitude determination. *Journal of the Astronautical Sciences*, 25(3):773–783, 1977.

[8] F. G. Valdes, L. E. Campusano, J. D. Velasquez, and P. B. Stetson. Focas automatic catalog matching algorithms. *Publications of the Astronomical Society of the Pacific*, 107:1119–1128, 1995.

[9] T. Dzamba, J. Enright, D. Sinclair, K. Amankwah, R. Votel, I. Jovanovic, and G. McVittie. Success by 1000 improvements: Flight qualification of the st-16 star tracker. *28th Annual AIAA/USU Conference on Small Satellites*, 2014.

[10] K. M. Huffman. Designing star trackers to meet micro-satellite requirements. Master's thesis, Massachusetts Institute of Technology, Cambridge, 2006.

[11] Goodrich Corporation Optical. Space systems – star trackers. `https://web.archive.org/web/20080517004222/http://www.isr.goodrich.com/StarTrackers.shtml`. 2019-05-19.

[12] T. Sands. Phase lag elimination at all frequencies for full state estimation of spacecraft attitude. *Physics Journal*, 3:1–12, 12 2017.

[13] Sinclair Interplanetary. *Second Generation Star Tracker (ST-16RT2)*, 2017. Rev. 2017a.

[14] Jena-Optronik. *Autonomous Star Sensor ASTRO 15*, 2015.

[15] Jena-Optronik. *Autonomous Star Sensor ASTRO APS*, 2015.

[16] Hyperion Technologies. *ST400 Star Tracker*, July 2018.

[17] Hyperion Technologies. *ST200 Star Tracker*, July 2018.

[18] Jena-Optronik. *ASTROgyro^{TM}*, 2015.

[19] W. W. Weiss, S. M. Rucinski, A. F. J. Moffat, A. Schwarzenberg-Czerny, O. F. Koudelka, C. C. Grant, R. E. Zee, R. Kuschnig, St. Mochnacki, J. M. Matthews, P. Orleanski, A. Pamyatnykh, A. Pigulski, J. Alves, M. Guedel, G. Handler, G. A. Wade, K. Zwintz,, the CCD, and Photometry Tiger Teams. Brite-constellation: Nanosatellites for precision photometry of bright stars. *Publications of the Astronomical Society of the Pacific*, 126:573–585, 2014.

[20] V. A. Muruganandan, J. H. Park, S. Lee, I.-S. Jeung, S. Kim, and G. Ju. Development of the arcsecond pico star tracker (apst). *Transactions of the Japan Society for Aeronautical and Space Sciences*, 60(6):355–365, 2017.

[21] Sinclair Interplanetary. Star trackers. `http://www.sinclairinterplanetary.com/startrackers`. 2019-05-19.

[22] ON Semiconductor. *MT9P031 1/2.5-Inch 5 Mp CMOS Digital Image Sensor*, 1 2017. Rev. 10.

[23] Marshall Electronics, Inc. *V-4416.0-1.2-HR 16.0mm F1.2 High Resolution CCTV Lens*, 2013.

[24] Altera Corporation. *Cyclone IV Device Handbook, Volume 1*, 3 2016.

[25] Integrated Silicon Solution, Inc. *IS42 / 45R86400D / 16320D / 32160D IS42 / 45S86400D / 16320D / 32160D 16Mx32, 32Mx16, 64Mx8 512Mb SDRAM*, 5 2015. Rev. B.

[26] Cypress Semiconductor Corporation. *CY15B104Q 4-Mbit (512 K 8) Serial (SPI) F-RAM*, 9 2015. Rev. C.

[27] Micron Technology, Inc. *MT25QL256ABA Micron Serial NOR Flash Memory*, 7 2018. Rev. K.

[28] STMicroelectronics. *STM32F401xD STM32F401xE ARM Cortex -M4 32b MCU+FPU, 105 DMIPS, 512KB Flash/96KB RAM, 11 TIMs, 1 ADC, 11 comm. interfaces*, 1 2015. Rev. 3.

[29] A. Slavinskis, U. Kvell, E. Kulu, I. Sünter, H. Kuuste, S. Lätt, K. Voormansik, and M. Noorma. High spin rate magnetic controller for nanosatellites. *Acta Astronautica*, 95:218–226, 2014.

[30] A. R. Ayal. Star detection algorithm for ESTCube-2 star tracker. Bachelor's thesis, University of Tartu, Tartu, 2016.

[31] M. Lindh. Development and implementation of star tracker electronics. Master's thesis, KTH Royal Institute of Technology, Stockholm, 2014.

[32] B. D. Warner. *A Practical Guide to Lightcurve Photometry and Analysis*, chapter Extinction and Transform Observations, pages 93–102. Springer, 2006.

[33] T. Dzamba, J. Enright, and D. Sinclair. Characterization of chromatic effects in small star trackers. *30th Annual AIAA/USU Conference on Small Satellites*, 2016.

Appendix A. Star Tracker Test Image Obtaining Experiment Setup

Test images were obtained at Tartu Observatory, Tõravere, Estonia ($58.265368°N$, $26.463807°E$) between February 24th, 2019 at 23:57 and February 25th, 2019 at 03:08. The following list describes the parameters of images taken. G stands for gain, t for exposure time, blc for black level compensation automatic calibration target.

For noise estimation:

- 25 bias frames, $G = 1, blc = 50$.
- 25 bias frames, $G = 8, blc = 50$.
- 25 bias frames, $G = 15.75, blc = 100$.
- 20 dark frames, $G = 1, t = 0.1s, blc = 50$.
- 20 dark frames, $G = 8, t = 0.1s, blc = 50$.
- 20 dark frames, $G = 15.75, t = 0.1s, blc = 100$.
- 20 dark frames, $G = 1, t = 1.0s, blc = 50$.
- 20 dark frames, $G = 8, t = 1.0s, blc = 50$.
- 20 dark frames, $G = 15.75, t = 1.0s, blc = 100$.
- 20 dark frames, $G = 1, t = 10.0s, blc = 50$.
- 20 dark frames, $G = 8, t = 10.0s, blc = 50$.
- 20 dark frames, $G = 15.75, t = 10.0s, blc = 100$.

For estimating the greatest magnitude stars that star tracker optics is capable of detecting:

- 5 frames of the Ursa Major (zenith), $G = 1, t = 0.1s, blc = 50$.
- 5 frames of the Ursa Major (zenith), $G = 8, t = 0.1s, blc = 50$.
- 5 frames of the Ursa Major (zenith), $G = 15.75, t = 0.1s, blc = 100$.
- 5 frames of the Ursa Major (zenith), $G = 1, t = 1.0s, blc = 50$.
- 5 frames of the Ursa Major (zenith), $G = 8, t = 1.0s, blc = 50$.

- 5 frames of the Ursa Major (zenith), $G = 15.75, t = 1.0s, blc = 100$.
- 5 frames of the Ursa Major (zenith), $G = 1, t = 10.0s, blc = 50$.
- 5 frames of the Ursa Major (zenith), $G = 8, t = 10.0s, blc = 50$.
- 5 frames of the Ursa Major (zenith), $G = 15.75, t = 10.0s, blc = 100$.

For air mass estimation over a short amount of time (12 minutes):

- 5 frames in the direction of Ursa Major α and β (zenith), $G = 8.0, t = 1.0s, blc = 100$.
- 5 frames in the direction of Perseus, $G = 8.0, t = 1.0s, blc = 100$.
- 5 frames in the direction of α Canis Minoris, $G = 8.0, t = 1.0s, blc = 100$.
- 5 frames in the direction of Gemini, $G = 8.0, t = 1.0s, blc = 100$.
- 5 frames in the direction of Leo, $G = 8.0, t = 1.0s, blc = 100$.
- 5 frames in the direction of Ursa Major ζ and η, $G = 8.0, t = 1.0s, blc = 100$.

Weather information for this data collection time and place:

- no clouds, fog or precipitation;
- visible distance 25 to 35 km;
- air temperature 1.4 to 2.0 degrees Celsius;
- humidity 84 to 90%;
- air pressure 1018.0 to 1021.0 hPa;
- wind speed 1.9 to 3.8 m/s.

Source: the Estonian Weather Service (`http://www.ilmateenistus.ee/`)

Non-Exclusive Licence to Reproduce Thesis and Make Thesis Public

I, Sandra Schumann

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   "Using Star Identification Algorithms on ESTCube-2 Star Tracker"

   supervised by MSc Hendrik Ehrpais and MSc Tõnis Eenmãe

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Sandra Schumann
20.05.2019