
MSc
Thesis
Automated Satellite Track Detection and Endpoint Determination in Astronomical Images
W.N.J. Rood
Technische Universiteit Delft
TU Delft

MSc Thesis

Automated Satellite Track Detection and Endpoint Determination in Astronomical Images

by

W.N.J. Rood

Student number: 4973739
Date : 20-09-2021
Email: w.n.j.rood@student.tudelft.nl
Supervisors: Dr. ir. Ernst Schrama TU Delft
Dr. Thomas Wijnen TU Delft

Cover image from FOTOS testdata, a stack of all images over one night of the center camera in La Silla (Chile). Highlighting all the detectable features for orbit determination.

Abstract

The Royal Netherlands Airforce (RNLAF) is currently supporting the Feasibility study for Optical Tracking of Orbital Satellites (FOTOS), which aims to create a satellite observation instrument, FOTOS1, for the optical tracking of satellites, which would be a key tool towards maintaining a sustainable use of space. This thesis was tasked with producing an improved image processing pipeline with the aim of detecting more objects and accurate endpoints for orbit determination. Additionally, it was important for the pipeline to maintain the cost and time efficiency of FOTOS1.

For time and cost efficiency, data was firstly reduced; the data used originated from the MASCARA instrument in Chile, which takes subsequent exposures at a fixed exposure length from dusk until dawn. The instrument consists of five cameras that virtually covers the whole local horizon. The number of frames to be processed varies depending on the time of year, but is in the range of $23,500$ to $34,000$ images. Since optical satellite tracking requires that the satellites are sunlit, we could reduce the number of images for processing by specifying an observable altitude limit of $h = 2,000$ kilometers. This reduced ranges from 4.35% up to 48.38% depending on the day of the year. For higher altitude limits the reduction becomes significantly smaller as the earth-shadow becomes less of an effect.

To increase the detection performance of the pipeline we combined 50 images to create a single detection image. Several operations were performed to highlight dynamic features among the starry background. The astrometric solution was used to align the images together and subtract subsequent images from one another. These difference images were stacked together by their maximum value to highlight longer streaking features. For the detection, the Probabilistic Hough Transform worked efficiently and returned the correct positions on the image frame. The created detection images did not lead to an increase in the number of uniquely detected objects, but it did extend the altitude range of detected objects and the number of endpoints per unique object. Since the features in the stacks are longer the detection method was able to detect objects up to GEO altitude. Also since the stacks consisted of 50 images, a single detection of a feature could contain data from multiple images and thus contain multiple endpoints. Because of this approach the number of endpoints per unique object was increased almost fourfold whilst only detecting 20% less unique objects in total.

Our last task was to determine the endpoints within one track; two novel methods were produced and tested with the aim to use position and discrete time data to create an overall better representation. However, it turned out that fitting noisy data was difficult for the tested regressors (least squares, Theil-Sen and RANSAC). It oftentimes caused the determinations to be off by 15 or more pixels- ruling those results useless for orbit determination. Both the index prediction and index tracing methods had trouble defining the shape due to noise and thus can benefit from a possible iterative approach for data selection. The double index prediction method combined with a Theil-Sen regressor was selected for the pipeline as it showed to be the best performing method being the most robust combination and had relatively little error compared to the other combinations. When comparing the best performing new method to the existing method, it turned out that the existing method performed better. The endpoint accuracy of the new method was centered around 2 pixels but was more distributed than the existing methods. However the quantity of endpoints was almost doubled. How these results translate into quality of orbit determination is a recommended topic for future work.

During this thesis project we therefore produced a new pipeline which can be implemented for instruments with the same observation strategy as MASCARA and is compatible with different lenses and exposure times. It showed proof-of-concept that stacking images and through the data reduction simulation allowed for the processing of all the images within 24 hours such that a backlog of data will be avoided.

Acknowledgements

I want to thank my daily supervisor Thomas Wijnen, who's feedback, time and expertise has been invaluable to this thesis. I am grateful for the guidance and support from Ernst Schrama, whose input was fundamental in this project. I want to express my appreciativeness for Bernd Rieger who made himself available to me during a busy period for key insights into the world of image processing. I also want to extend a special thanks to Remko the other members of the FOTOS team for their feedback and inspiration from our bi-weekly meetings.

My final thank you is for Karel and Rini who generously hosted me for the duration of my studies at TU Delft.

List of abbreviations

Abbreviation	Full
AS	Astrometric solution
ASBG	All stack background
ATMC	All Track Mask Clipped
bRing	B Pictoris project
CPU	Central processing unit
DL	Detection Length
DIP	Double Index poly-fit endpoint Prediction
ESO	European Southern Observatory
FoV	Field of view
FOTOS	Feasibility study for Optical Tracking of Orbital Satellites
GEO	Geostationary Orbit
GSE	Geocentric solar ecliptic reference frame
HT	Hough Transform
IOD	Initial Orbit Determination
LEO	Low Earth Orbit
LS	La Silla (station in Chile)
LTT	Light travel time
MAE	Mean Absolute Error
MASCARA	Multi-site All-Sky CAmeRA
MLG	Maximum Line Gap
MLL	Minimum Line Length
PDF	Probability Density Function
PHT	Probabilistic Hough Transform
PIT	Poly-fit Index Trace endpoint determination
PSF	Point Spread Function
RANSAC	Random Sample Consensus (also used as RS)
RNLAF	Royal Netherlands Airforce
RT	Radon Transform
SA	South Africa, bRing station location in South Africa
SCT	Sigma clipped threshold
SGP4	Simplified General Perturbations propagator
TEME	True equator mean equinox inertial reference frame
TLE	Two Line Element
UTC	Coordinated Universal Time
VTL	Valid Track Length

Frequently used constants

Description	Symbol	Value	Unit	Source
Earth equatorial radius	$R_{\oplus}$	$6378,136$	m	[34]
Earth gravitational parameter	$\mu_{\oplus}$	$3.986004418 \cdot 10^{14}$	$m^3 s^{-2}$	[34]
Earth sidereal day	$T_{\oplus}$	$86,164.1004$	s	[34]
Astronomical Unit	AU	$1.4959787066 \cdot 10^{11}$	m	[34]
Sun Radius	R_{sun}	$6.957 \cdot 10^{8}$	m	[15]
Speed of light	c	$299,792,458$	ms^{-1}	[34]

Contents

1 Introduction

This chapter presents the current issues of maintaining a sustainable space domain and how the aim of this research is to provide an improved pipeline for the optical tracking of orbital satellites.

1.1. Motivation and problem statement

Due to the ever increasing use of space for commercial as well as military applications, the Royal Netherlands Airforce (RNLAF) will require the possibility of observing satellites for space situational awareness. Due to the continuous decrease in launch costs, and the recent advancements made in hardware engineering, the barrier for placing satellites in space is decreasing. Consequently, the space domain is becoming increasingly occupied. Guidelines for maintaining a sustainable space domain are formatted, but not enforced as of now. The threat is that this situation can possibly lead to the Kessler syndrome [12]. The Kessler syndrome states that with a higher number of objects in space there exists a risk of a potential collision. Where the debris from a collision causes a 'snowball effect' of more collisions. The end result is that only debris remains, making that domain of space unusable for anyone to use. Detecting and classifying objects in space is paramount in avoiding such events taking place, thus any contribution to this effort is beneficial to a more sustainable use of space. As for the military aspect, the detection and identification of classified satellites is desired for possible intelligence applications.

The Feasibility study for Optical Tracking of Orbital Satellites (FOTOS, [35]) was performed for the purpose of better detection and classification of objects in space. Its results from this study confirmed that it is possible to automatically detect sunlit satellites in astronomical images, and perform an initial orbit determination (IOD). Their method, FOTOS1, makes use of the fact that satellites can still be lit by the sun, whilst the observer on Earth is in the dark. In the images, stars will appear stationary whilst the satellites will leave a streaklet. These streaklets can be determined, where the endpoints of each streaklet serve as locations in time for determining from which orbit the streaklet originates from. In the FOTOS project the Gauss method was used for IOD, where three unique positions can translate to an approximate orbit.

The data used for this research are from two astronomical star surveys with a wide field of view (FoV). The image sources are the MASCARA en bRing instruments. Both instruments take images with a fixed cadence, causing an amount of data that is not feasible to process manually. In order to make full use of this data, the images need to be processed automatically with minimum amount of human intervention as possible . The need is expressed that the to be developed pipeline processes the data of the preceding night during the day before the next observations starts. This way, any given station does not create a backlog of data. The processing pipeline will have to be both efficient and flexible with the incoming data to account for different station configurations and amounts of images available.

The flexibility is also required in terms of observation strategy. Potential changes to the lens and observation cadence can be changed if needed with little changes in the written pipeline routines. For instance, if the cameras have a smaller FoV or the exposure times are set to be longer.

Lastly, detection and endpoint estimation methods need to be developed such that they reliably

and robustly create data points for orbit determination. The need exists for more accurate endpoint determination, but also more endpoints per unique objects. More accurate endpoints increase the quality of the orbit determination, and more endpoints allow for the use of batched orbit determination methods.

This concludes the general motivation and problem statement of the research. In the next section, the research question for this thesis is defined and the sub-goals to obtain the answer are specified.

1.2. Research question and objectives

The main goal for this research is to exploit the MASCARA and bRing instruments for their characteristic observation strategy; due to their high and continuous cadence in image capturing, and the availability of an astrometric solution (AS), we can create images that contain longer satellite features without drift in the star background. This creates a composite image of the night sky with a longer timespan, with accurate discrete information of the satellites position over time. Due to the amount of data that is created by this observation process, there exists a need to develop a processing pipeline that processes the data automatically and efficiently. The main research question can be formatted as follows:

To develop an efficient automated satellite track detection algorithm that is able to detect satellite tracks of objects in the Low Earth Orbit (LEO) domain and possibly higher, by way of stacking images, image processing operations, feature detecting methods and fitting the data with an overall fit, to be able to more robustly and accurately, determine the endpoints of streaks and thus in-directly improve the performance of the orbit determination algorithms.

This description encapsulates everything that needs to be done. However to achieve the result, several sub-goals need to be achieved. These are split into distinguishable goals upon which also the chapters will be formatted.

The first sub-goal is to reduce the incoming data such that only images that are more likely to contain satellite streaklets are stored. Since the satellites need to be sunlit, the altitude of the object defines how and when an object remain sunlit relative to the observer. To reduce the data that needs to be processed, we can perform a simulation of the observations to determine what frames are more likely to contain streaklets against time. This relation depends on the targeted object altitude, station location and time of year. This sub-goal can be defined as:

For each station, date and satellite altitude range, what are the useful images for each night such that only images are considered are most likely to contain sun-lit objects and thus satellite streaklets?

As there are multiple stations on Earth, the pipeline is required to be flexible depending on a station's location. The station's location and the day of the year correspond to the length of the night and thus what number of images to be processed. The altitude of the satellite can be used as a limit as it is the most important parameter for being sunlit or not.

The second sub-goal is to process the data itself and look for the expected features. For efficient processing the images can be combined by stacking them together and processed to account for possible visual effects of the stacking method. The features that are left by the satellites need to be highlighted for the chosen detection method. The goal can be defined as:

What is the optimal method for combining and processing the astronomical images for robustly detecting any satellite feature, taking into account the potential shape of the expected features?

This sub-goal can be split up into three aspects; image processing, feature analysis and feature detection. This process is sequential since each step is necessary for the next. In other words, a successful detection method is ultimately dependent on how the two former steps were performed.

Post feature detection, the third sub-goal is to determine the endpoints of the features. To do this we aim to make use of the information from using multiple images together, as there is then also information in the time domain (although discrete). The current FOTOS1 method was a 'local method', i.e. it only worked for a single difference image that consisted of two streaks. This means only information of position was used to determine the endpoint, and no time. The new methods are expected to determine the endpoints more accurately as they use more information, however the possibility of a bias due to the methods can be introduced. The sub-goal is summarized as:

After the features have been detected, what is the best method for determining the endpoints by using information from multiple discrete moments in time to achieve an overall better fit with with greater accuracy and negligible adverse effects on robustness?

Since the images do not contain information of time besides the exposure time and the moment at which it was taken, no reliable info can be gathered from the features between the endpoints of the streaks. The methods that will have to be investigated should be able to deal with the discrete representation of time, both in terms of resolution and outliers from other sources or faint features. The methods will be compared against the existing local method and the positions according to a database.

1.3. Novelty of the research

The novelty of our image processing pipeline is that we create a combined image of 320 seconds, with greater detail for objects with varying brightness. This is because we combine several images together to create a single image, a so called stack. To combine the images we use the stars as a reference so that they remain stationary. The aim is to succesfully detect fainter objects against a background containing less stars. The difference between this method and conventional astronomical observations are the following:

- There is no need for sidereal star tracking as the exposure time is set such that stars remain stationary and can be aligned with other images by using the stars as a reference.
- The stacked image has a longer passage time whilst keeping the star brightness equal to that of the exposure time. For longer exposures with sidereal star tracking (stars remain stationary), the stars would become the brightest feature on the image and saturate the sensor. This approach is not new for astronomical applications but is novel for the application of satellite detection.
- For longer exposure times no use can be made of the intermediate positions of the satellites in the image, however the stacked image contains discrete information in time by checking the corresponding source image.

There is an ongoing study that is being done with the OWL observatory where the researchers have tried to use a chopper wheel in front of the camera to gain information in-time [20]. However, this application caused issues with timing as the matching of time proved to be a difficult challenge. This study will work with better quality time stamps, however much less frequent than the OWL study. The frequency of the OWL study is done for up to 50Hz (0.02 seconds) whilst this study will work with 6.4 seconds only.

2

Input Products

To get started we need data to work with, this chapter will discuss the products that are used to detect satellites and determine their endpoints. These products are generated by two instruments which will be covered first, after which the details of the products themselves are discussed.

2.1. Instruments

The instruments that are available for this research are MASCARA and bRing. Both these instruments were designed for scientific research, one for exoplanet detection and the other for the large Hill sphere of β-Pictoris. Whilst their scientific goals are different, their observation strategy is similar to a certain degree. The instrument details and the characteristics useful for this research are discussed next.

2.1.1. MASCARA

The Multi-site All-Sky CAmeRA (MASCARA) is an exoplanet detection instrument that observes the brightness variation of stars during the night [28]. The variation in brightness of stars over time can be used to determine if an exoplanet is possibly orbiting the host star. The dip in observed brightness over time can be used to determine the orbital period of the exoplanet and give an indication of its mass. This method of exoplanet detection is called the ”transit-method”. To perform this research a long term data set is required, as well as a decent frequency of observations to increase the observational accuracy of the brightness variation.

The instrument consists of five cameras that each captures images angled in a cardinal direction (north, west, south and east) or zenith (straight overhead). Each camera is equipped with a 24mm lens,resulting in a FoV of 53×74 degrees. The combined FoV almost results in a perfect dome-like coverage of the sky above. The details on the pointing of the cameras is available in table 2.1. The goal of MASCARA is clearly to observe the stars, so it is crucial that the stars remain stationary in the images. The usual practice for astronomers is to track the objects over time with a mount. However, since MASCARA monitors a big portion of the sky, it is easier to set the exposure time low enough such that the stars don’t drift in the images. The exposure time of 6.4 seconds was chosen to be used since; it is an integer multiple of seconds in a sidereal day, it suits the research objective of MASCARA in terms of star magnitude (brightness) and pixel saturation, and that it is low enough such that stars don’t appear as trails in the images. A interline CCD [28] sensor is used by MASCARA and it allows for a continuous observation strategy. The beneift of this is that whilst one exposure is being taken, the previous one can be read out.

One of the issues that all cameras have is the so-called Point Spread Function (PSF). The PSF defines how an observed bright point source object is translated to the image. This is especially the case for satellites, as the objects themselves are significantly smaller than the pixel size, however their perceived shape is multiple pixels wide due to the PSF. The PSF for all the cameras of the MASCARA instrument are visible in [28].

There are two MASCARA stations currently operating; one in La Palma Spain (LP) and one in La Silla Chile (LS). Currently, only the data from the LS station is available as the La Palma station is under

Table 2.1: The theoretical pointing directions of each of the cameras in the MASCARA configuration from the MASCARA software package (an old *.pro* file). In reality, the installation of the cameras can be altered to avoid conflicts with objects in view. The orientation of 0 degrees corresponds to a landscape frame, 90 degrees is rotated to portrait mode. For the La Silla the station itself was rotated 10 degrees to avoid the dome of another nearby telescope on site being within one of the cameras' FoV [28].

Camera	Direction	Altitude [deg]	Azimuth [deg]	Orientation [deg]
C	Center	0	0	0
N	North	63.5	0	90
E	East	53	90	90
S	South	63.5	180	90
W	West	53	270	90

Table 2.2: Instrument locations for MASCARA and bRing along with their geodetic coordinates in degrees and elevation in meters. Also the time correction for the coordinated universal time (UTC) is stated. UTC does not have a daylight savings correction and is thus consistent year round.

Instrument	Site ID	Cameras	ϕ, λ [deg]	Elevation [m]	UTC [hrs]
MASCARA	LP	5	28.7621, -17.8777	2396	1
MASCARA	LS	5	-29.2611, -70.7314	2400	-4
bRing	AU	2	−31.2722, 149.0622	1798	2
bRing	SA	2	−32.3812, 20.8102	1165	11

operation of a different institute. Further details about the positioning of the stations can be found in table 2.2

2.1.2. bRing

The bRing stations were designed to observe the development of the β-Pictoris Hill-sphere [27]. Their design has some overlap with MASCARA as it also observes the background stars with the same cadence and exposure time. However, due to the scientific goal of bRing only the even frame numbers are of the 6.4 second exposure type, whilst the odd exposures are 2.54 seconds. This difference causes the observation strategy to be somewhat different as there are no subsequent exposures; Instead of continuous tracks there exists equitemporal gaps between the streaklets. Since the objective of bRing is to observe β-Pictoris, the pointing of the stations is optimized for keeping the star in view across both stations, meaning they both point somewhat southwards as the declination of the star is -51 degrees. The pointing itself table 2.3) is split into East and West to distinguish between the two cameras. The two stations combined are capable of observing the star for the majority of 24 hours, where MASCARA in La Silla is able to close the full 24 hour cycle.

2.2. Data

The data used from MASCARA and bRing consists of astronomical images of the sky and the AS for accurately overlaying subsequent images. The AS uses a catalog of stars as a reference to create a solution for accurately overlaying the images over each other and determining the exact position of each pixel in the celestial sphere. The applications of the AS for this research are; aligning subsequent images and translating pixel positions to celestial coordinates. An application that is already used by MASCARA is to create an 'difference image' representing the difference between two subsequent images, which can then be used to detect a potential transit. These difference images are also useful for detecting dynamic objects like satellites. In a single difference image a satellite will leave one positive streak (the reference image), and one negative streak (the subtracted image).

Table 2.3: Approximate theoretical pointing of the South Africa bRing instrument (SA). The altitude and azimuth of the cameras are correct, the orientation is an approximation. Information taken from [27].

Camera	Direction	Altitude [deg]	Azimuth [deg]	Orientation [deg]
E	East	45	150	60
W	West	45	210	60

Table 2.4: Observation duration for each station for the longest and shortest nights. The observation duration is translated to an expected number of observations based on eq. (2.1). The start and end times are based on the civil twilight (sun 12 degrees below horizon). The times are given by www.timeanddate.com, where the station coordinates from table 2.2 are used for positioning. Times are denoted in hh:mm format.

Station	Date	Start	End	Duration	Images	Stacks
LS	20-06-2021	18:48	06:40	11:52	33375	668
LS	21-12-2021	21:42	05:40	07:58	22406	448
LP	21-06-2021	22:12	06:14	08:02	22594	452
LP	21-12-2021	19:14	07:04	11:50	33281	666
SA	21-06-2021	18:37	06:40	12:03	6778	136
SA	21-06-2021	20:46	04:23	07:37	4284	86
AU	21-06-2021	18:05	06:05	12:00	6750	135
AU	22-12-2021	21:09	04:54	07:45	4359	87

Table 2.5: Most important and used camera parameters, taken from [27, 28]

Parameter [unit]	Figure
Field of view [deg]	53x74
Sensor size [px]	2672x4008
Pixel scale [1] [deg/px]	0.01983532934
Exposure time [s]	6.3825

2.2.1. Quantity of observations

The observation strategy for MASCARA and bRing is that they make their observations whilst the sun is 10 degrees below the horizon. Based on the day of the year the start and end-time of observation can be determined. From this an approximation can be made for the amount of images that will be taken in the ideal case (clear skies). For this the times T are in seconds.

$$N_{frames} = \frac{T_{end} - T_{start}}{6.4} N_{cameras} C_{station} \tag{2.1}$$

The 6.4 represents the 6.4 second cadence of the images, $N_{cameras}$ is the number of cameras of the station and $C_{station}$ is a coefficient of 1 or 0.5 dependent on the station. Since we are only interested in the 6.4 second exposure, we only need the even exposure. Thus, the 0.5 correction needs to be applied for the bRing stations as it it takes two different exposure times (6.4 and 2.54). For the four stations, approximations are made for the start and end times with civil twilight, i.e. when the sun is 12 degrees below the horizon. This means that the estimates made are conservative.

The sheer quantity of images expected requires efficient processing. For best case this equals to almost one second per image for both detection and endpoint determination, but also orbit determination. By using the ability to stack the images together can speed up the detection step. Also being able to filter out images based on target altitude beforehand is beneficial.

2.2.2. Images

The images from the cameras are given in the *.fits* format, which is a common application in astronomy. The sensor size of the cameras used is 2672×4008 pixels, meaning each pixel corresponds to about 0.02 degrees in the FoV. The pixel conversion is dependent on the position on the frame, the common transformation between the two is one arcminute (0.01667 degrees). Specifics of the cameras used that are later applied in the pipeline are mentioned in table 2.5. The most important parameter of the images is that the exposure time of both cameras are 6.4 seconds (6.3825 seconds exactly). The bRing station's exposure time of 2.54 seconds is not compatible with the 6.4 second exposures as the signal strengths are different. Since the majority of data available consists of the 6.4 second type, the methods are developed to this exposure time. The 6.4 second exposure is chosen such that the star drift in the images does not exceed the pixel scale, meaning that stars remain stationary in each image.

Besides the regular science frames, each camera also take calibration frames at the start of their routine. These frames are used to remove and or reduce noise. Although these frames are taken, they weren't used in the current MASCARA processing pipeline due to negligible gain in optical quality. This

Table 2.6: All the available testing data for this research. Consisting of data from the two bRing stations and the La Silla MASCARA station. The camera identifier is the two letter station abbreviation and the camera orientation identifier (La Silla Center = LSC).

Date	Camera	Files	Delta	Science	Difference	Bias	Darks	Flats	Used
20191114	AUE	5128	4927	4928	0	0	200	0	
20191115	AUE	5112	4911	4912	0	0	200	0	
20200103	AUE	4828	4627	4628	0	0	200	0	
20200104	AUE	4836	4635	4636	0	0	200	0	
20200102	LSC	4765	4739	4645	95	40	40	40	
20200103	LSC	4771	4745	4651	95	40	40	40	x
20200104	LSC	121	0	1	0	40	40	40	
20200102	LSE	4765	4739	4645	95	40	40	40	
20200103	LSE	4771	4745	4651	95	40	40	40	x
20200104	LSE	61	0	1	0	20	20	20	
20200102	LSN	4765	4739	4645	95	40	40	40	
20200103	LSN	4771	4745	4651	95	40	40	40	x
20200104	LSN	61	0	1	0	20	20	20	
20200102	LSS	3511	4739	3391	1349	40	40	40	
20200103	LSS	4771	4745	4651	95	40	40	40	x
20200104	LSS	61	0	1	0	20	20	20	
20200102	LSW	4594	4739	4474	266	40	40	40	
20200103	LSW	4771	4745	4651	95	40	40	40	x
20200104	LSW	81	0	1	0	40	20	20	
20191114	SAE	3688	3487	3488	0	0	200	0	
20191115	SAE	5048	4849	4848	2	0	200	0	
20191121	SAE	4956	4755	4756	0	0	200	0	
20200103	SAE	3889	3689	3689	1	0	200	0	
20200104	SAE	4768	4569	4568	2	0	200	0	

is because the dome of the instrument does not close properly and thus there exists some bleeding of light from the outdoors, which makes them unreliable. There are three types of calibration frames available; darks, bias and flat frames. In chapter 4 the application is discussed.

For the testing of the processing pipeline a few folders of testdata are available. However not all testdata is useful since not all nights of observations are complete. Most importantly, the data for all the West cameras of the bRing stations are missing, and for the La Silla MASCARA station only one night where all cameras have the same number of images is available. All the testing data is formatted in table 2.6.

Within the testdata there are different frames available. Besides the calibration frames, there are also the regular 6.4 second exposure image which we call science frames. The science frames have an identifier based on the exposure number. The difference between the highest and lowest exposure number we call the delta of the testdata folder. With this figure we can estimate if there are any frames missing in a folder, as the number of science frames should be equal to the delta of the science frames. From table 2.6 it is clear that the best data set available is the LS station for the 3rd of January 2020: set is consistent among all the cameras, meaning that they share the number of exposures. The only thing lacking for that data set is the last exposure in one stack, this is always the exposure that ends on 49. Therefore, we chose to use this night of observations for the development of the methods, as it is consistent among cameras and nearly complete. It means that all analysis performed in this thesis is done with the data set of MASCARA La Silla 2020-01-03.

Besides the images themselves the *.fits* file also contains header data. In the header, information is stored about the image, for instance the exact exposure time or the date. The data that is used from the header is mentioned in table 2.7. The most important parameters are the Local Sidereal Time (LST) and the Julian Day (JD). These are both used for correctly timing the exposures and thus making sure that the positioning of the satellites are as accurately as possible.

Besides the data for timing, other parameters are also extracted for analysis. The reference and

Table 2.7: Header data that is extracted and used in further processing steps.

Header key	Description
LST	Local sidereal time in hrs at image midpoint
JD	Julian date at midpoint of exposure (unmodified)
LSTSEQ	Reference sidereal time sequence number
X0	Start coordinate of active region
Y0	Start coordinate of active region
XSIZE	Width of active region (frame size)
YSIZE	Height of active region (frame size)
EXPTIMES	Set exposure time in seconds (reference)
EXPTIME	Measured exposure time in seconds

real exposure time are compared to see if the camera is operating consistently. Also the dimensions of the frame are taken from the header of the file such that images with alternative dimensions can be processed too. The other remaining parameters that are mentioned in table 2.7 are used as parameters in the subtraction routine of MASCARA.

2.2.3. Astrometric Solution

A very important product for this research is the AS, as we use this product for aligning a group of images to the same star reference background. Astrometry is defined as *performing precise measurements of positions and movement of stars and other celestial bodies in the sky* [1]. For MASCARA and bRing, the AS is mainly used for converting the sky position to pixel position (and vice versa). This can then be used to; accurately subtract two subsequent images from each other, bin the images together in a stack for data reduction, or determine the pixel position of the satellites from a catalog. For this research we use those principles, but we combine the first two together to create detection images. We first create difference images from subsequent images to highlight the streaks, and then combine those difference images together to create a stack, essentially creating longer features. We will use the the maximum group size, fifty images, as the number of frames for creating the stacked images (however bRing only has 25 exposures of 6.4 seconds). This limit of fifty comes from the MASCARA and bRing pipelines as the AS was created for each group of fifty images. The reference position for an image group is the middle exposure time of the group. MASCARA provides an even number of images, thus the midpoint of the group is set as the 26th in the list. For bRing stations the midpoint is better defined as the 13th in the list, i.e. exactly in the middle. The alignment accuracy of the AS within one group has a standard deviation of about 0.2 pixels for the cardinal directions and 0.3 pixels for the zenith camera [28]. This means that with the $3-\sigma$ rule we can confidently say that the alignment error is well within one pixel for 99.73% of the images.

2.2.4. Two-line Element catalog

For determining the position of satellites in the frame we make use of a Two line element (TLE) catalog. TLE's are the public standard for tracking objects in space [31]. One TLE includes, but is not limited to, information about the objects orbit and launch date. It is important that the catalog stems from the same date as the testdata. For all the available testdata, we have the TLE catalogs from *Space-Track*.[2] If we would apply the TLE catalog of today for a year-old data set, the positions of the objects would not match due to decay and or drift that occurred during the year. All objects in space experience perturbations that causes change in their orbits, meaning the orbit of objects continuously change. The scale of change is dependent on the orbit itself, for instance lower objects experience more drag due to the thin atmosphere. The changes of these orbits also need to be continuously cataloged, which again shows the relevance of tracking objects continuously.

2.3. Concluding

For this thesis, the analysis is done on the data available from the MASCARA station in La Silla Chile on the 3rd of January 2020, because it is the only night of observations that is nearly perfect when

[2] https://www.space-track.org

compared to the other testdata. The research will however be performed such that it is applicable to any station type and any location. After this research only a few parameters would have to be updated by the user such that the algorithms are compatible with different inputs.

3

Data Reduction

As mentioned in section 2.2.1 there is a significant amount of images that need to be processed. This quantity of images can be reduced by evaluating if it is possible to observe sunlit satellites in these images. This is done by simulating the position of the observer overnight and checking what portion of the night-sky is both visible (per camera) and sunlit (per altitude). The altitude is represented in a discrete manner by a number of points that is translatable to the resolution of the cameras. The altitude is selected such that it represents the upper boundary of the object range, meaning if $2,000$ kilometers is selected, the visibility of objects from zero to $2,000$ kilometers is estimated for. Next, the points are evaluated if they are visible to the observer and if they are lit by the sun. If both of these conditions are satisfied it is possible to give an estimate on how useful each frame is for detecting satellites up to the specified altitude. The simulation is done in the Geocentric Solar Ecliptic reference frame (GSE), this makes it so that the altitude points are stationary with respect to the sun. The GSE frame is shown in fig. 3.1.

This simplifies the calculation since the estimation for the sunlit condition only has to be performed once. For the simulation, we take as the following as inputs: the station location (e.g. La Silla), station type (MASCARA/bRing), date, the upper altitude limit and a threshold for when images can be neglected. As an output the simulation gives a so-called blacklist of images that are not of direct interest. This describes the general working of the simulation, whilst the respective components of the simulation will be discussed in detail further. Lastly, the results from the simulation will be discussed and validated against detection results from the existing detection algorithm.

3.1. Time

This simulation will be performed for the specified day of the data available, i.e. the date of the evening of observations (the start). The observations are then taken up to the morning of the next day. The time step that is chosen is equal to the exposure time of the instruments, meaning a full day constitutes 13.500 points in time for which the simulation has to be evaluated. To make sure that the observation start and observation ends are within the simulation time range, the simulation runs from 12:00 local time of the observation day to 12:00 local time the next day (24 hours). For this time range the suns altitude is estimated and if this is 10 degrees below the observers horizon (astronomical twilight), the simulation can continue. Within the simulation the date and time are represented in JD, where the integer value represents the date and the numbers after the decimal point represents the progress of time for the given day in Universal Time (UT). The simulation time thus runs from $JD + 0.5$ up to $JD + 1.5$ plus an extra correctional term for the timezone that the stations are in. The conversion of the input date(YYYY-MM-DD) to JD was taken from an existing Github repository [1].

3.2. Observer

As mentioned previously, the simulation is done in the GSE reference frame. We first assume that the earth is a perfect sphere instead of an ellipsoid, with a radius of 6378 kilometers (equatorial radius).

[1] https://gist.github.com/jiffyclub/1294443, *jdutil.py*, version: Oct 18, 2011.

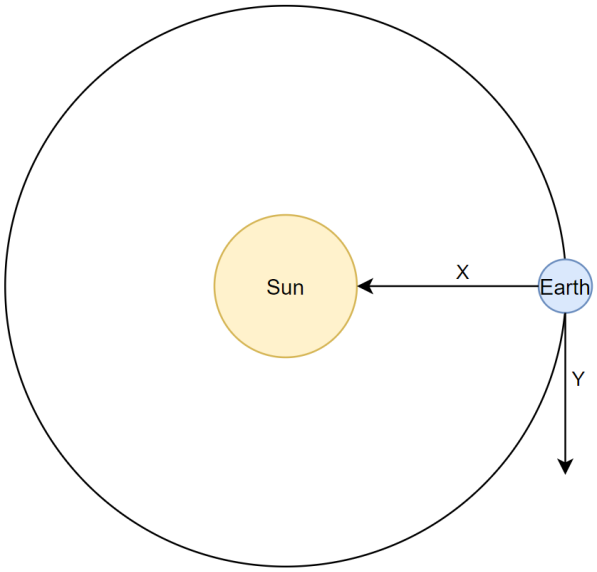


Figure 3.1: Simple visualization of the Geocentric Solar Ecliptic (GSE) reference frame. The x-axis always points towards the sun from the earth, where the z-axis is perpendicular to the orbital plane of the earth around the sun (the ecliptic) and pointing north. The y-axis then points approximately opposite of the earth's velocity vector. [9]

Because of this assumption the observer position in the simulation will be different to reality. Both the position (in elevation) and the orientation of the observer on the earth's surface are not exactly the same. The radial error is however small (0.34%) and will be taken into account in the results. The position of the observer over time is determined with relations from [9]. The observer details mentioned in chapter 2 are given in latitude(λ), longitude(ϕ) and elevation (h), and are converted to a Cartesian state (x,y,z) in the Earth-Centered, Earth-Fixed reference frame (ECEF). This is done with eq. (3.1), where $R_{\oplus}$ denotes the earth's equatorial radius.

$$\begin{aligned} x &= \left(h + R_{\oplus}\right) \cos\phi \cos\lambda \\ y &= \left(h + R_{\oplus}\right) \cos\phi \sin\lambda \\ z &= \left(h + R_{\oplus}\right) \sin\phi \end{aligned} \tag{3.1}$$

From that frame (ECEF) there are three rotations that need to be applied to obtain the observer's position in the simulation frame (GSE). One rotation that defines the position of the observer on a rotating earth (around the polar z-axis), another one that translates the rotation of the earth around the sun (over the ecliptic) and lastly one that translates the earth's equator to the plane on which the earth revolves around the sun (ecliptic). As an input to these transformations we need the time representation in Universal Time (UT) and Julian Centuries (JC, or epoch 2000.0), this we can obtain by using eq. (3.2) from the user guide for "Space physics coordinate transformations" [9].

$$JC = \frac{JD - 24515450}{36525.0} \tag{3.2}$$

In the user guide the transformation to go from ECEF to GSE is defined as:

$$\overline{X}_{GSE} = \mathbf{T}_2 \mathbf{T}_1^{-1} \overline{X}_{ECEF} \tag{3.3}$$

Where the Cartesian state vectors are denoted by $\overline{X}$ and the two transformation matrices are denoted by $\mathbf{T}$. Where the $\mathbf{T}_2$ matrix consists of two rotations. Also the negative exponent for the $\mathbf{T}_1$ matrix indicates a rotation in the opposite direction. The representation of the respective rotation matrices are as follows (following the format of [9]). The first transformation matrix represents the rotation from fixed earth to inertial earth.

$$\mathbf{T}_1 = \langle \theta, Z \rangle \tag{3.4}$$

Where the angle θ is given in degrees by an expression in the user guide [9]:

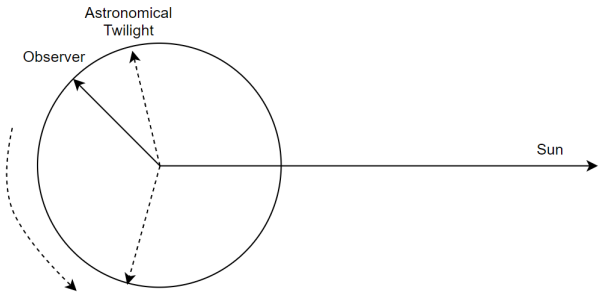


Figure 3.2: 2D representation of the observer position with respect to the sun direction (x-axis). Note that the position of the observer changes in 3D and thus can also move inwards into the circle for movements up and down.

$$\theta = 100.461 + 36000.770JC + 15.04107UT \tag{3.5}$$

The second transformation matrix consists of two rotations; one rotation to go from the earth's equator to the ecliptic of the earth-sun system, and a rotation that fixes the sun's position for the new frame.

$$\mathbf{T}_2 = \langle \lambda_\odot, Z \rangle * \langle \varepsilon, X \rangle \tag{3.6}$$

Again the respective angles in degrees are given by an expression in the user guide [9]:

$$\varepsilon = 23.439 - 0.013JC \tag{3.7}$$

$$\begin{aligned} M &= 357.528 + 35999.050JC + 0.04107\text{UT} \\ \Lambda &= 280.460 + 36000.772JC + 0.04107\text{UT} \\ \lambda_\odot &= \Lambda + (1.915 - 0.0048JC)\sin M + 0.020 \sin 2M \end{aligned} \tag{3.8}$$

With the observer's position defined in time, we are also able to compute the start and end of the observation window. When the angle between the x-axis (sun direction in GSE) and the observer is greater than 100 degrees (10 degrees below the horizon), the observation start is triggered. This is visualized in a 2D representation in fig. 3.2.

The angle between two vectors can be calculated with the dot product formula

$$\cos\theta = \frac{u \cdot v}{\|u\|\|v\|} \tag{3.9}$$

Since the sun direction is a unit vector in the direction of the x-axis this expression can be simplified to:

$$\theta = acos\left(\frac{\overline{X}_{observer,x}}{\left\|\overline{X}_{observer}\right\|}\right) \rightarrow \text{if } \theta \geq 100 \text{ degrees} \tag{3.10}$$

This condition is checked at every timestep and as mentioned before serves as a trigger to continue further calculations. The next step is to accurately define the FoV of the observer and its respective cameras.

3.3. Observer Cameras

The coordinate system that is used for the observer is the horizontal coordinate system. The reference system defines the sphere around the observer, where the position of objects are given in the angles altitude (or elevation) and azimuth. The distance to the target can also be given but is not used in this application as the objects are at an arbitrary distance. The coordinate system is shown in fig. 3.3.

For the simulation we use the MASCARA instrument which has five cameras pointing in each of the cardinal directions and zenith (overhead). To determine what each camera sees, we need the pointing

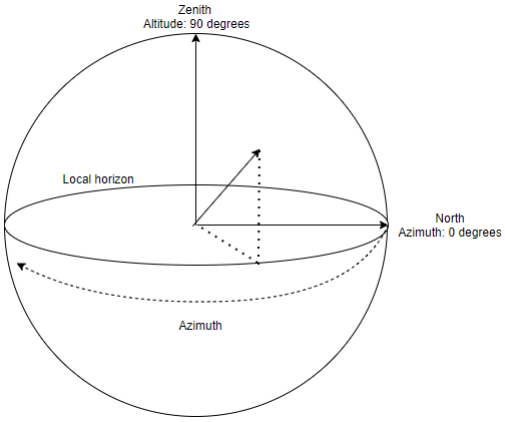


Figure 3.3: The coordinate frame used for determining the pointing directions of the cameras.

direction of each camera. This consists of the altitude angle, the azimuth angle of each camera and the orientation of the cameras around the pointing axis. These are stated in table 2.1. To determine the extent of the cameras we use adapted functions from the supplied MASCARA environment to create a discrete border representation of the FoV of each of the cameras. The FoV of the complete MASCARA instrument in Chile is shown in fig. 3.4.

To determine whether there are points in view of a camera, the altitude angle and azimuth angle of each point is computed. Then by using a function available in the Python package *matplotlib* [3] we can determine if a point is within the polygon representation of the projected sky. After determining which points are within each camera's view, we need to determine what percentage of those points is sunlit. This can then give an estimate for the usefulness of the frame for satellite detection.

3.4. Satellites

The satellites are represented in a discrete manner by using an approximation for equidistant spacing of points on a sphere [5]. This approximation divides the sphere into latitude and longitude domains $(d\phi, d\vartheta)$, and aims to satisfy an equal distribution of area on the sphere depending on the given number of points. Instead of the N_{points} input of the paper ([5]), we can adapt it such that it approximates a certain angular accuracy. This is preferable, since it is easier to translate the pixel representation of the cameras to the number of points of the discrete sphere. It is known that the MASCARA and bRing instruments have an angular resolution of about 0.02 degrees. If a discretisation of $1point = 10\,pixels$ would be used, the sphere would consist of just about over a million points ($N_{points} = 1,030,860$). A finer representation of the angular resolution will lead to more accurate results, however the computation time drastically increases and causes the simulation to crash due to memory storage constraints. A table which compares the angular resolution with Central Processing Unit (CPU) time and other key results can be found in section 3.6 The radius of the sphere is determined by the h_{limit} that is specified. Only the upper limit of the altitude range is simulated as it defines the outer extent of what the system wants to detect, meaning that if a frame is not interesting for objects at 2,000 kilometers, it is definitely not interesting for lower objects. If also lower altitudes are weighed into the simulation, the cutoff for when frames are regarded as useless is obtained earlier. This should be avoided as frames might still be useful for objects at altitudes near the specified h_{limit}. For the altitude we use the earth's equatorial radius of 6378.136 kilometers taken from [34]. The radius for the discrete sphere is therefore:

$$R_{satellite} = R_{earth} + h_{limit} \tag{3.11}$$

3.5. Condition Calculations

The simulation starts by creating the discrete sphere for the given altitude limit described in section 3.4. All these points are then checked if they are sunlit or not. Multiple conditions are checked such that the computation is done efficiently, these are as follows:

- Points with positive x-values are all sunlit as they are on the sun-side of the earth in the GSE

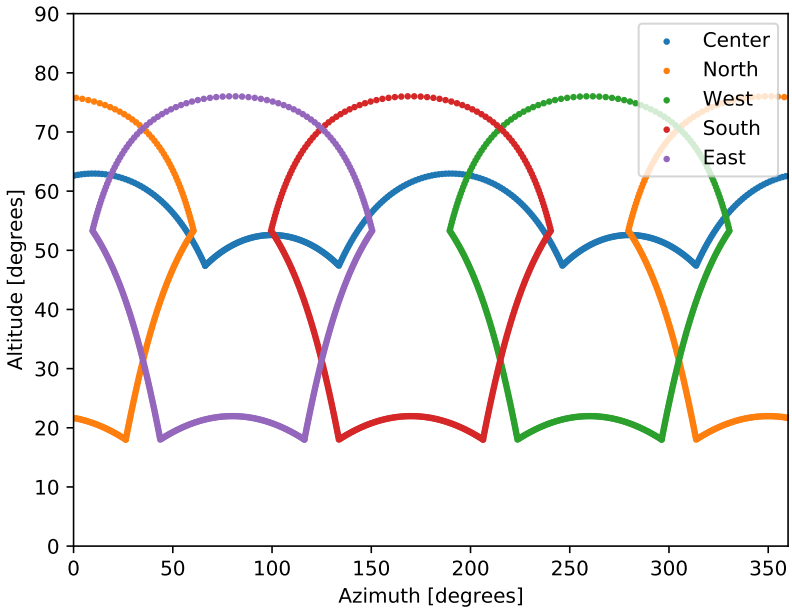


Figure 3.4: Representation of the camera FoVs of the MASCARA instrument in La Silla (Chile). Note that; the Northern camera is split into two sections as it passes through the boundary where 360 degrees wraps back around to zero degrees, the cardinal direction cameras are rotated 10 degrees around the nadir axis, the center camera FoV covers the complete region of altitudes up 90 degrees. The exact pointing of the cameras was obtained through discussion with the one of the instrument's investigators.

reference frame (x-axis points towards the sun). These sunlit points are visible in the first region of fig. 3.5. The remainder of the points (with negative x-values) are further processed.

- Of the remaining points it is determined if the hypotenuse of the y and z components is greater than the earth radius. If this condition is true, the point is beyond the perimeter of the earth and therefore also sunlit. Points that are smaller than this are further evaluated.

- The remaining points are projected on the yz-plane and scaled such that they represent the outer edge of the earth ($x = 0$). These perimeter points are then subtracted from the original points, which leaves only the relative vector from the earth's perimeter to the point itself. The angle that this vector makes with the x-axis is used to determine if the point is in the umbra or penumbra region behind earth. The Umbra and Penumbra regions play a role if objects are sunlit or not. As sometimes the light coming from the outer radius of the sun wraps around the earth, causing a region behind earth that is pseudo sunlit (penumbra). The calculation is shown in fig. 3.6 and the umbra and penumbra regions are shown in fig. 3.7. The step-by-step calculation is given next.

Now we define the operations needed to check the last condition that is mentioned. We define the satellite point as $\overline{R}_{satellite}$, this vector is then projected onto the yz-plane. To create the projection onto the yz-plane, we create a projection of $\overline{R}_{satellite}$ onto the x-axis, and subtract that from $\overline{R}_{satellite}$. This is denoted as:

$$proj_{yz}\left(\overline{R}_{satellite}\right) = \overline{R}_{satellite} - proj_{x}\left(\overline{R}_{satellite}\right) \tag{3.12}$$

We then obtain the perimeter position for this satellite point by scaling the projected vector such that the norm is equal to the earth's equatorial radius.

$$\overline{R}_{perimeter} = R_{\oplus} \cdot \frac{proj_{yz}\left(\overline{R}_{satellite}\right)}{\left\| proj_{yz}\left(\overline{R}_{satellite}\right) \right\|} \tag{3.13}$$

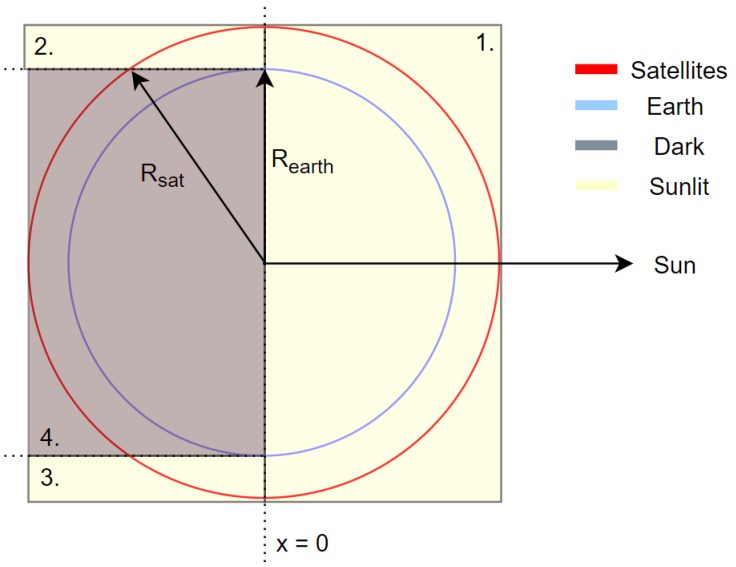


Figure 3.5: 2D visualization of the simulation. The circles denote the earth and the given satellite limit. In this figure, the sun is positioned at infinity. Region 1 defines the points that are sunlit since they are on the sun-side of the earth. The Region 2 and 3 are sunlit since they exceed the earth radius in y and z. Region 4 contains the points that are not sunlit.

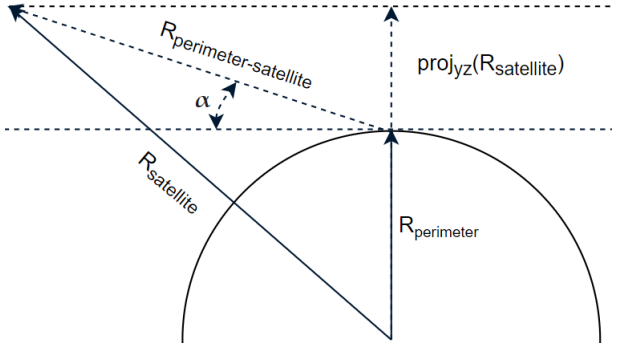


Figure 3.6: The visualization for determining what angle (α) the satellite points make with the outer perimeter of the earth.

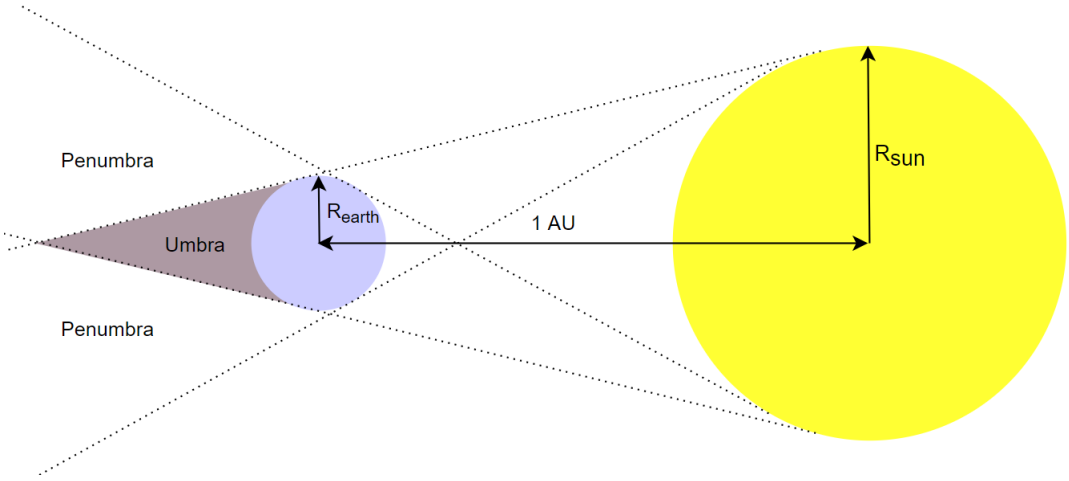


Figure 3.7: The umbra and penumbra regions behind the earth. The distance between the earth and the sun is one Astronomical Unit (AU). Note that this figure is not to scale as the distance between the earth and sun is two orders of magnitude larger than the sun's diameter.

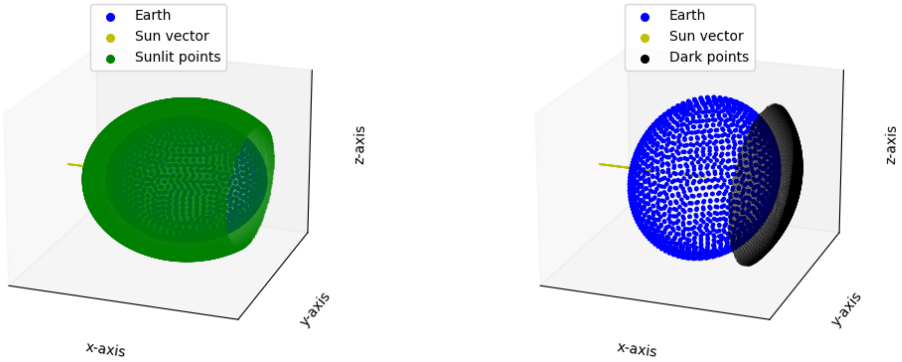


Figure 3.8: The sunlit and dark points of the discrete satellite sphere shown in two sub-figures. The limiting altitude is set to 2000 kilometers (LEO) and the angular resolution to 1 degree.

The vector that runs from the perimeter to the satellite is defined as:

$$\overline{R}_{perimeter-satellite} = \overline{R}_{satellite} - \overline{R}_{perimeter} \tag{3.14}$$

The angle this vector makes with the horizontal x-axis can be used to determine if the satellite is occulted from the sun by the earth. This can again be simplified since we compare only with the x-axis, similar to what was done in eq. (3.10).

$$\alpha_{sat} = acos\left(\frac{\overline{R}_{perimeter-satellite,x}}{\left\|\overline{R}_{perimeter-satellite}\right\|}\right) \tag{3.15}$$

The angles that the sun's perimeter makes with the earth's perimeter are split up between the umbra and penumbra angles. The umbra angle defines the dotted lines in fig. 3.7 that cross each other behind the earth, and the penumbra angle is defined by the dotted lines that cross in-between the earth and sun. These angles are calculated by:

$$\alpha_{umbra} = atan\left(\frac{R_{sun}-R_{earth}}{AU}\right) = -0.264\,degrees \tag{3.16}$$

To determine if a satellite is sunlit we only have to check the condition for the umbra angle, as we only determine if there are point in the penumbra, and are thus pseudo sunlit. Since it is unknown what the perceived brightness will be of the satellite in the penumbra compared to being "fully" sunlit, it is assumed that objects within the penumbra are sunlit (to be safe). To determine if the remaining points are in the penumbra or in the umbra we only have to check if the the angle (α_{sat}) is less than the umbra angle. Also since the penumbra angle is only 0.05 degrees from the umbra angle it does not have to be taken into account in the simulation due to the restricted resolution.

$$satellite = \begin{cases} Umbra & \alpha_{sat} < \alpha_{umbra} \\ sunlit\,(penumbra) & \alpha_{sat} > \alpha_{umbra} \end{cases} \tag{3.17}$$

Since the angular resolution of the cameras is approximately 0.02 degrees, and it is infeasible for the simulation to run with this accuracy, it seems that distinguishing between umbra and penumbra does not add anything substantial. It is, however, implemented in case there exists a clear relation between brightness and regions in the penumbra. This is something that can be investigated after the detection stage- depending on the detection results it can be beneficial to compare object brightness to sky position. The resulting sunlit and dark points of the satellite sphere is shown in fig. 3.8.

After the sunlit and dark points have been determined, the next calculation is determining the position of the observer over time. Next, we determine the angle between the observer vector and the sun vector (x-axis) as is stated in eq. (3.10). If this condition is satisfied, the local horizon of the observer is evaluated and the points within the FoV of the observer. This is done by evaluating the angle between the following vectors:

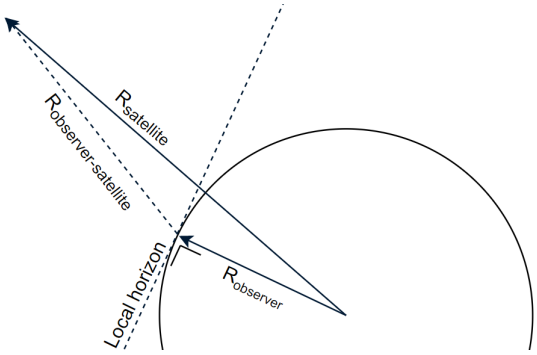


Figure 3.9: 2D overview for determining the local horizon of the observer and which points are in the FoV of the observer.

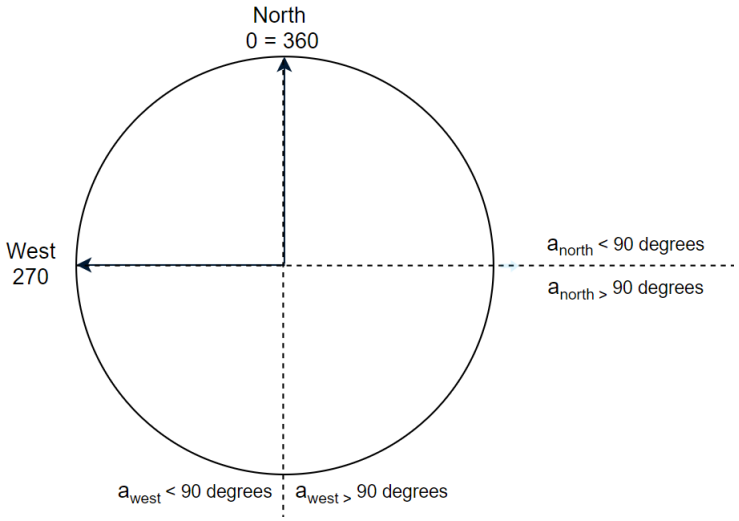


Figure 3.10: Visualization of determining the azimuth angle by using the two reference local angles (north, west). The four quadrants for the azimuth are denoted by the dotted lines. By using the angles between the horizontal (north) and vertical (west) axes, we can determine the quadrant of the vector.

- The vector that runs from the frame origin (earth) to the observer: $\overline{R}_{observer}$
- The vectors between the observer and all the altitude points: $\overline{R}_{observer-satellite}$

The angle between the two vectors we define as $\alpha_{altitude}$ as it defines the visual height of the point above the horizon of the observer. If this angle is greater than 90 degrees (local horizon), we know that the points are within the FoV of the observer. This is graphically shown in fig. 3.9. In the local coordinate system the height above the horizon is known as elevation and is thus 0 at the horizon and 90 degrees at zenith, which is slightly different. Points within the FoV are checked if they fit within the FoV of a specific camera. For this we still need to determine the azimuth angle for each point. This is done by creating local North and West vectors. The local North vector is created by defining the polar north in the ECEF frame (which is $[x = 0, y = 0, z = R_{earth}]$), and the West vector is created by performing the cross product of the observer vector and the North vector. These vectors are converted to the GSE frame in the same way as the observer was converted. The angles that each point make with the local North and local West vectors are used to define the azimuth angle. For this we need conditional statements as the angles for north and west can be on either side. This is defined in eq. (3.18) and shown in fig. 3.10. The result for defining which points are in which camera is shown in fig. 3.11.

$$\alpha_{azimuth} = \begin{cases} \alpha_{sat,north} & \alpha_{sat,west} > 90 \\ 360 - \alpha_{sat,north} & \alpha_{sat,west} < 90 \end{cases} \tag{3.18}$$

After all the points have been evaluated, a percentage of sunlit points for each camera and moment in time is determined. A threshold is defined when a frame can be regarded as uninteresting. This is

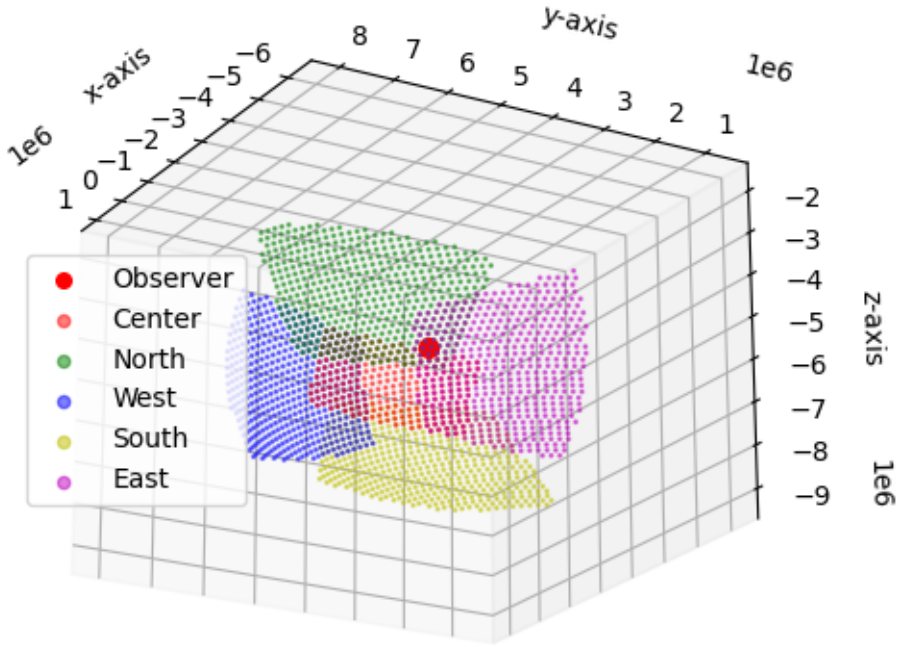


Figure 3.11: Visible points per camera for an arbitrary point in time in the reduction simulation for 2020-01-03 La Silla (Chile). The overlap between cameras can be seen since the opacity of the points is set at 50%. The angular resolution was set to 1 degree.

currently set at 10% of the points being sunlit, but can be adjusted to the user's liking. The percentages per camera over time for one of the testdata nights is plotted in fig. 3.12.

The figure above illustrates that the testdata night of 2020-01-03 contains a period during the night in which there are no sunlit points in the FoV of the camera pointing in the northern direction. The image groups (stacks) containing only images below the threshold are then discarded from further investigation. It means that even if there is one frame above the cutoff percentage in a group of 50 images, the group is still kept for further processing. This is done to be safe in terms of missing potential objects of interest. Further results from the simulation will be discussed next.

3.6. Results

The expectation was that when given a limiting observing altitude, we are able to reduce the data to be processed by checking the percentage of sunlit points per camera. For a limiting altitude of $2,000$ kilometers we can obtain the potential reduction of images to be processed for the longest and shortest night for the MASCARA station in La Silla (Chile), which is presented in table 3.1.

Table 3.1: Data reduction results for the longest and shortest nights for the La Silla MASCARA station in Chile. As Chile is in the southern hemisphere, the longest and shortest nights are flipped to those in Europe. Note that the number of images is according to the simulation. The resolution that was used is 1 degree per pixel.

Date [-]	$\mathbf{h_{lim}}$ [km]	Images			Reduction [%]
		Available [-]	Useful [-]	Discarded [-]	
2020-12-21	2,000	23,490	22,469	1,021	4.35
2020-12-21	6,000	23,490	23,490	0	0.00
2020-06-21	2,000	34,310	17,712	16,598	48.38
2020-06-21	6,000	34,310	30,707	3,603	10.50

Table 3.1 confirms that the availability of frames increases as the night gets longer. However, the

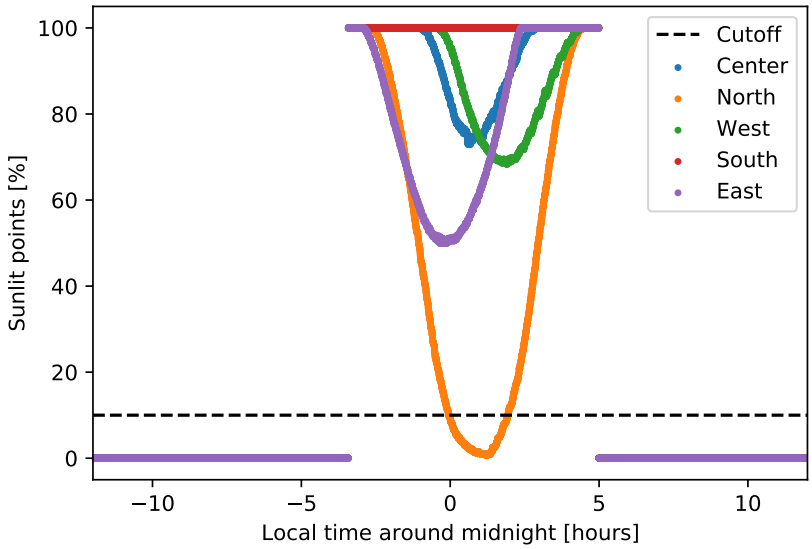


Figure 3.12: The history for the percentage of sunlit points per camera over time for the night of 2020-01-03 La Silla (Chile) for an altitude limit of $2,000$ kilometers. The cutoff is at 10%

usefulness of those images might not necessarily be better for the LEO domain, as the number of useful frames decreases for longer nights. In terms of the aforementioned goal of the data reduction routine, it is good that the reduction for the longest night is close to 50% for the set altitude limit. However, if a limit is chosen that is beyond LEO (e.g. $6,000$ kilometers), the effectiveness of this routine decreases by 80% with respect to the $2,000$ kilometer limit. This means that the reduction decreases significantly. Depending on the detection performance this limit can be defined. However the reduction in images might become so small that it does not make sense to exclude any images.

3.6.1. Performance and accuracy

As mentioned in section 3.4, the angular resolution needs to be chosen for the discrete sphere. For multiple accuracies the simulation was performed on the the 2020-01-03 testdata night. The aim was to see if increasing the angular resolution will lead to more accurate results, and if the CPU time remains do-able. The results are detailed in table 3.2.

Table 3.2: Results for performing the reduction simulation for multiple angular resolutions. Note that the last column contains both the percentage of frames to be discarded and the absolute number of real images that can be discarded.

Ang. res. [deg]	Pixels per point [-]	$\mathbf{N_{points}}$ [-]	CPU time [s]	Discarded images [%]
10.00	500	404	15.08	6.47
5.00	250	1,632	18.61	5.12
2.00	100	10,270	46.48	4.63
1.00	50	41,164	143.41	4.88
0.50	25	164,828	545.17	4.76
0.20	10	1,030,860	3,336.06	4.78

As can be derived from the results shown in the table above it is clear that increasing the angular resolution causes the simulation to be a lot more CPU intensive. Of course the simulation can be altered for some more performance, by for instance using a C++ compiler [13]. In terms of accuracy the need for a better angular resolution only stretches to about 1 degree. Any greater accuracy will lead to extra computational effort that might be unnecessary since we discard stack-wise and not image-wise. Therefore we use an angular resolution of 1 degree for this thesis.

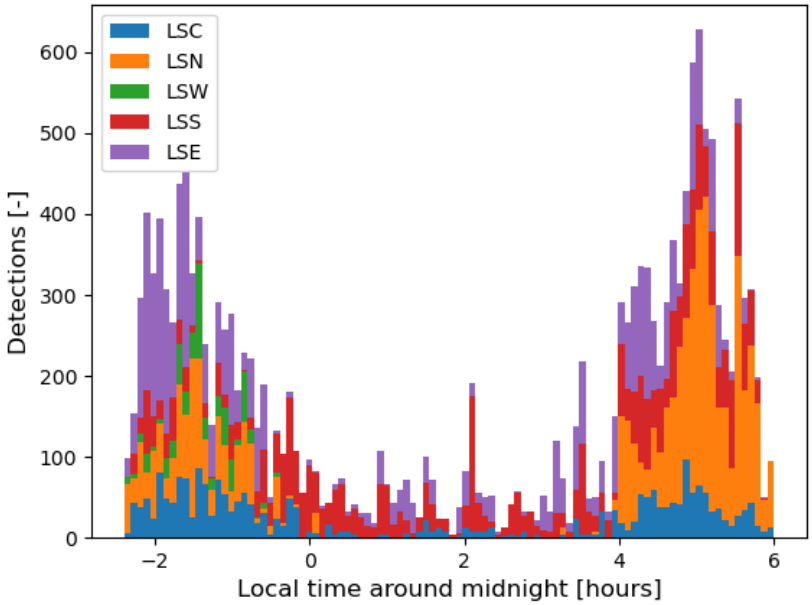


Figure 3.13: Detections by the existing FOTOS1 routine for all the LEO objects in the testdata available from the La Silla MASCARA station. The detection time represents the time of day, i.e. the detections of multiple nights can be grouped together.

3.6.2. Validation with detection data

To see if the simulation correlates with real-world data, we can compare detections of the existing FOTOS1 routine against the results from the simulation. To compare against fig. 3.12, we use all the detections from FOTOS1 of objects that are lower than $2,000$ kilometers in altitude. The existing FOTOS1 routine was developed for single difference images and thus only covers 12.8 seconds per image. It means that the method is by default only capable of detecting low passing objects as these leave longer features. There are however detections of objects with higher altitudes, but these are, for the sake of this comparison, omitted, but brought up in discussions later in chapter 6. The LEO detections of the two full nights of testdata are binned together to create a single histogram. The histogram shares the x-axis like the one in fig. 3.12 to make comparisons possible.

What is evident is that the majority of detections happen in the evening (dusk) and in the morning (dawn). This is to be expected as the lower altitude objects are less likely to be sunlit in the middle of the night. This result correlates well with the simulation as is visualized in fig. 3.12, which shows dips in sunlit percentage in the middle of the night. Especially the dip in the northern camera percentage is confirmed by the detections, as there are close to zero detections in the middle of the night. This makes sense, as the northern direction in the southern hemisphere looks towards the darkest patch of sky behind earth (please see fig. 3.8 for further details). Also the continuous detections of the southern camera during the night match well with the simulation. From the simulation it was also evident that there should be a bias in the east and west cameras at sunset and sunrise. Since the sun sets in the west, it is to be expected that the western camera should detect more objects at dusk, and vice versa (eastern camera and dawn). This does come forward in the simulation (fig. 3.12), but does not show clearly in the detections. At closer inspection it turned out that during the sunset a bright moon was present in the images. This caused the detections from FOTOS1 to be dominated by false positive detections whilst also decreasing the likelihood of a good detection. This explains why there is less detections at the beginning of the night compared to the end of the night.

3.6.3. Discussion

The results of the data reduction simulation correspond well for the LEO object range, and can also give a good reduction of data to be processed. The need for this simulation, however, becomes less relevant for higher altitudes as the reduction becomes significantly less for these objects. Depending on

Figure 3.14: Visualization of the phase angle between the source (sun), the observer (earth) and the object. The gradient on the object shows the brightness that is coming from the source. The observer is only able to see part of the sunlit side of the object, causing a drop in perceived brightness.

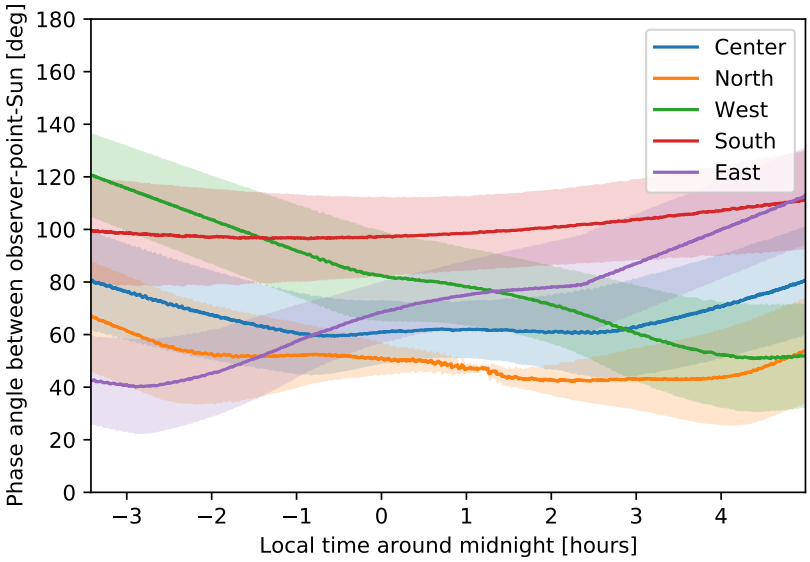


Figure 3.15: The average phase angle of sunlit points within each camera during the night. The region around the average is equal to one standard deviation of the points.

how well the new detection method will work in terms of object altitude, the application of this reduction simulation might become obsolete. If it turns out that there are detections during the whole night, the simulation can be altered for a different application. If, for instance, a single camera were to be used, the simulation can return the optimal viewing direction for detecting satellites. Alternatively, the simulation could be used to estimate what type of features we should be expecting in each image. As for each image, the altitude for sunlit objects change and thus also the features change.

Another aspect from the simulation is that it is unknown what the perceived brightness will be of the objects in view. Multiple arbitrary parameters decide if the satellite will be bright enough to be detectable by the cameras (e.g. object size, object material). One of the factors that decides this brightness is the phase angle between the sun, observer and the object. The phase angle defines the angle of at which the object is lit by the source of light (the sun) with respect to the observer. When the source and observer are close to being aligned with each other the phase angle is small, meaning that the observer sees a larger portion of the object that is sunlit. This is shown in fig. 3.14.

For observing satellites and this simulation, the percentage of sunlit points and the phase angle of those points are competing objectives. As the percentage of sunlit points drops, the phase angle of the sunlit points is smaller. This is shown in fig. 3.15.

These findings illustrate that even though the percentage of sunlit points might be low, the possible brightness of these points is larger. It is therefore advised to not set the cutoff percentage too high as it might result into discarding images that still contain bright satellites.

The results found and discussed in this chapter is fundamental to decreasing computation time if there is such a need. By extension, this aids in reaching the first defined sub-goal. However, for the purpose of this thesis, no data from this night was omitted in further sections.

4 Image Processing

The standard input images are not directly useful for the detection of satellites. The input images have a value range of 16 bits ($[0, 65535]$), whilst the detection methods are able to deal with 8-bit ($[0, 255]$) images at most, and preferably binary ([0,1]). Intermediate steps are therefore required to go from the input images to detection images. For this research the aim is to exploit the feature of an AS to stack multiple images together to generate a single image representing a longer time-frame and thus contains longer features (as described in chapter 2). Stacking the images together is not as straightforward, since other features will cause issues that obscure the satellite tracks. In this chapter the methodology for creating the detection images from the input products will be described. This will be done by describing the image operations and showing intermediate products as well as flowcharts. The overall flowchart for going from the input products to the final detection image is shown in fig. 4.1. The steps that are mentioned in this flowchart serve as an outline for the sections in this chapter.

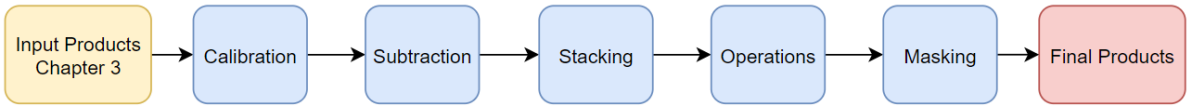


Figure 4.1: Flowchart for the steps to be taken to go from the input images to the detection images.

The existing algorithm (FOTOS1) used in the feasibility study was only applied to single difference images and not stacks of difference images. The working of this algorithm can be found in [35]. The FOTOS1 algorithm for highlighting the tracks is also applicable to the stacked images, however, post testing it was found that noise made the detection images useless. Therefore, there was still room for improvement for the FOTOS1 method. Also, the discrete time information that can be extracted from combining images can be applied in the image operations.

All the performed image operations and steps are described in this chapter and shown in appendix A. Full resolution images and figures can be found in the Gitlab repository. [1]

4.1. Image Calibration

Firstly, the input images need to be calibrated. Due to the age of the instrument and thus also the sensors, the number of hot pixels were significant. For single images this is not a problem, since these ”hot” pixels only leave a local feature. However if 50 images are aligned to the star reference, the hot pixels are shifted in position and leave a streaking feature. This is not wanted as it can confuse the line detection method later on. To remove these hot pixels we need to perform dark frame calibration [1]. Dark frames are created by taking exposures of a pitch-black environment. These images should in theory then only contain the noise profile of the sensor. By averaging multiple dark frames into one master dark frame, the noise readout of the sensor can be estimated more accurately. Especially hot pixels show up in the master dark frame since they exist in each frame.

[1] https://gitlab.strw.leidenuniv.nl/rood/fotos_python3, request access through email.

The MASCARA and bRing instruments both have 40 dark calibration frames for each night. These dark frames are combined by taking the median pixel value over all 40 dark frames. The median value is taken since this provides a more reliable estimate of the background which deals with possible outliers better. The master dark frame is subtracted from each input image, resulting in a calibrated image which is used further.

Besides the dark calibration frames for thermal noise there are also bias frames for readout noise. However the application of these frames did not lead to any improvement of the final product. It seemed that the bias frames also estimated the hot pixels quite well, but the extra application of bias frame calibration was deemed unnecessary.

4.2. Image Subtraction

Instead of stacking the images directly, we create so-called difference images first. This initial operation removes the majority of the brightest features and highlights the satellite streaks from the background. The difference images are made by subtracting two subsequent images from each other, which results into a composite image containing the brightness variation between the two respective input images. As mentioned in chapter 2, there are routines available in the MASCARA and bRing software packages for creating the difference images. As aforementioned in chapter 2, to create difference images we use the AS to align a group of images with each other, where the middle image serves as a positional reference. One difference image needs two input images, one of these images we call the positive frame (A) and the other image we call the negative frame (B). The B frame is subtracted from the A frame to create the difference image, hence the positive and negative definitions. This causes the A frame variation to be positive and the B frame variation to be negative. Furthermore, the pixels of a satellite streak present in the A frame will have positive values whilst the satellite streak pixels in the B frame will have negative values. In the existing subtraction routine from MASCARA and bRing the first image is subtracted from the second image, meaning that the B frame has an an index lower than the A frame. To create as many difference images as there are input images, we invert the values of the first difference image to transform the negative B frame into a positive A frame. This is unlike the subtraction routine of FOTOS1 where the absolute value of the difference images was used, creating two streaks per image. The function that describes this procedure is stated below:

$$\text{Difference}_i = \begin{cases} \text{image}_1 - \text{image}_2 & i = 1 \\ \text{image}_{i+1} - \text{image}_i & i => 1 \end{cases} \tag{4.1}$$

For more clarity, it is also included in a flowchart of the next section; fig. 4.2. Note that this procedure is only meant for exposures of equal length. Since bRing has different exposure lengths in subsequent images, the resulting product will not be the same. A shorter exposure will lead to a weaker signal, and if subtracted from a stronger signal image it will not be able to remove the features of stars and other objects. For bRing there exists a 6.4 second break in between the two potential satellite streaks. This is convinient since there is no risk of missing information due to subtraction of overlapping streaks. Since the satellite streaks are not lines of single pixel width, there exists a risk of subtracting information of the ending/starts of the streaks. This is a present risk in the MASCARA data, and is something that should be changed within the subtraction routine in the future, where an option would be to subtract only even and odd images from each other. The scale of the error that is induced depends on the visual brightness of the satellite, the alignment of the subsequent streaks, the PSF of the camera and the position on the frame.

4.3. Stacking

After the difference images are made we can combine them into making composite images, namely stacks. There are four different stacks that are made with the difference images; value stacks, even stacks, odd stacks and index stacks. The first three are created by using the maximum value from a set of difference images. For the value stack this is all the available difference images, and for the even and odd stacks it depends on the parity of the difference images. The application of the even and odd stacks will be explained in section 4.4. The index stack is different as it does not use the maximum value, but contains the index of the difference image the maximum value originates from. The function that is used is $argmax$, which is a *numpy* function that returns the source of the maximum value. The

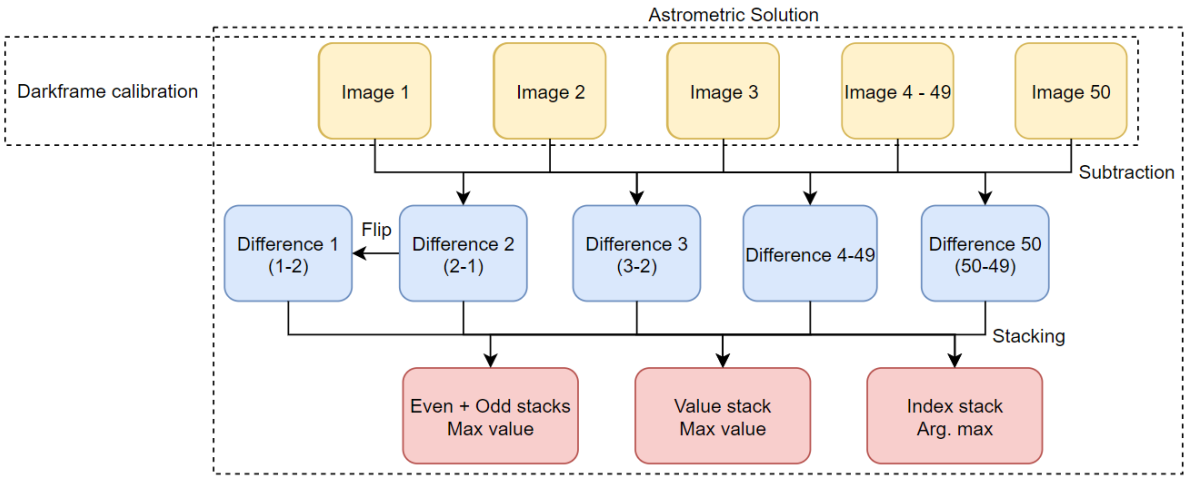


Figure 4.2: Procedure on how the stacks are made from the input products.

index stack discretely represents the time of the bright features, as it returns the integer of the source difference image (and each difference image has a timestamp from the positive A frame).

Stacks consist of a group of fifty difference images for MASCARA, which equals the size of the group within one AS. The maximum group size is selected because it maximizes the possibility to detect slower moving objects.

The number of groups available each night is approximately equal to dividing the number of frames available by 50. It does, however, depend on the index of the images themselves, as the AS is made for images that share the same quotient response (last two digits are within the range $[00 - 49]$). The modulo operation determines how many times a number fits into another number (quotient), and what the remainder is. For instance 7 fits 3 times in 23 (quotient = 3), and when subtracting $3 * 7$ from 23, the remainder is 2. Groups are made for images that share the same quotient when divided by 50, the remainder is then at most 49. This can cause the beginning and the end of the night to be incomplete [2]. For groups that have gaps in the image sequence, the currently approach is to use the next image in the queue for subtraction. However, there is still some optimization to be done when that happens, since the observation conditions might be different due to the time passed. The optimal solution is to create difference between those that have close timestamps. This is an optimization that still needs to be implemented in the subtraction routine.

The functions that describe the creation of the stacks is below. The subscript g denotes the group designator within the data set, the subscript i denotes the image index within the group. Note that for the difference images the index range started at 1 and now it starts at 0.

$$\text{Stacks}_g = \begin{bmatrix} \text{Value stack}_g & = max(Difference_i) & i = 0, 1, 2, .., 49 \\ \text{Even stack}_g & = max(Difference_i) & i = 0, 2, 4, .., 48 \\ \text{Odd stack}_g & = max(Difference_i) & i = 1, 3, 5, .., 49 \\ \text{Index stack}_g & = argmax(Difference_i) & i = 0, 1, 2, .., 49 \end{bmatrix} \tag{4.2}$$

The workflow for creating the stacks is shown in fig. 4.2

4.3.1. Value stacks

The result from the maximum value in the stacking algorithm gives an image that shows all the brightest variational features over a period of 320 seconds relative to the astrometric reference. The resulting stacks show moving objects relative to the stationary star background. Satellite tracks within these stacks often pass through the whole FoV of a camera, depending on the velocity of the object and thus the orbit of the object.

An issue that exists in the value stacks is that there exists some halo-esque features, as visible in fig. 4.3. These are caused by the alignment routine of the frames within one group [28]. It is quite

[2] Other causes of missing data can be breaks in observation due to weather circumstances or storage errors.

evident through visual inspection that it is a numerical error in the images since they show a clear grid-like shape that is equal to the size of the tiles in the alignment routine (32×32). This grid size stems from the AS routine in MASCARA and was chosen such that it can optimally align the stars within the images. The halo-esque features are later corrected for in the image operations section of this chapter section 4.4.

Another issue that is raised in the stacking process is that the images are stacked with the timestamp at the middle of the exposure. For the stars this is an optimal solution since the drift is only minimal over 3.2 seconds. However, for fast moving satellites the change in position over 3.2 seconds can be quite substantial. This can cause the beginning and ending of the streaks within the track to be misaligned by a few pixels. Besides the object's velocity, it also depends on the orientation of the object (orbital plane) with respect to the stars movement, and lastly on the looking direction of the camera. The misalignment of the streaks will later be shown in chapter 5.

4.3.2. Index stacks

Besides the maximum value stacks there also is another product; the index stack. This stacked image contains the index of the difference image where the maximum value came from, rather than the maximum value itself. The index of the difference image then corresponds to the positive A frame from the subtraction routine, as those are positive values.

This product is very useful since it highlights the satellite streaks in time. In addition, a benefit of this product is that it removes the virtually random brightness variations of stars in the stacks. The downside is that the range of the image (in pixel value) is only equal to the number of difference images that is used within the stack. For instance, for bRing the value range is $[0, 24]$, meaning that signal to noise ratio is low. For future work the index stacks are very promising for the application of machine learning for detection purposes.

4.4. Image Operations

This section discusses the operations that are performed to go from the stacked images to the detection images. The detection images are the final products that are used to search for satellite features.

4.4.1. Background Subtraction

Due to the astrometric alignment, the halo-esque features artifacts are present in the value stack images. These should be removed as they create a concentration of bright points in the frame. When a line detection algorithm passes over the frame, a cluster of bright(er) points can cause the detection algorithm to become confused and return a false positive. To remove this background, it needs to be estimated. There are several operations available, but the by far the best estimation is a composite product from all the available value stacks of one night. By taking the median value for every pixel over all the created value stacks, the result is a variation-free image of the background. It completely removes all the dynamic features such as star variations, bright moon passes, lens flares and satellite tracks. It even is able to capture some of the remnant hot pixel streaks within the stacks. It is also expected that the method will have little trouble with possible atmospheric features like clouds as those also move across the frame. The product is referred to as the "All Stack Background" (ASBG) and is presented in fig. 4.3. The ASBG is subtracted from the value stack image and the result is the ASBG Subtracted image.

The ASBG method for background estimation worked better than general background estimation methods available. Especially local background estimators for single images did not work well since the signal strength of the satellite tracks was often still visible in the background. If this was to be subtracted from the stacked image, the signal from the tracks would weaken. Another promising method was the iterative clipping method by [16], however this still was not able to pick up the halo-esque features. By using the median value over all the available images, this resulting background is a very reliable estimate since it is not sensitive to outliers. What it does require is that there are enough stacks to work with, meaning that the selected testdata set is ideal (even though a bright moon was present in the first few hours). If, for instance, there was cloud formation and a consistent bright moon the background might show some false features. If that is the case a possible version from the preceding night could be used.

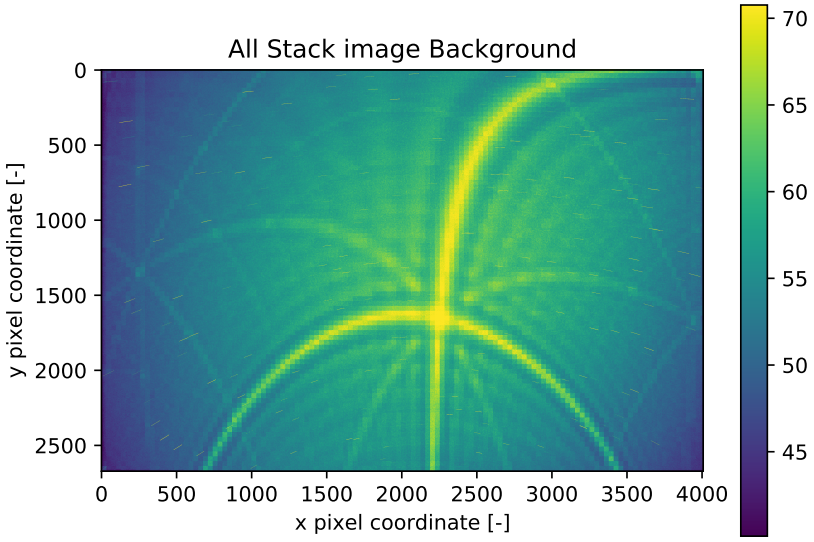


Figure 4.3: The All stack background for the La Silla Center (LSC) camera. The signal strength is shown in the colorbar.

4.4.2. Even-Odd Difference

An operation that was thought of during testing of background estimation methods was the Even-Odd difference operation. By splitting the images within the stack into two (even and odd), and stacking those sets into two separate products, both products should have about the same level of background noise and features. If we take the absolute difference of these two products, the resulting image should be free of background signal. For MASCARA this works very well as there are 25 images per parity stack, however the application for bRing still needs to be investigated as there are half the images to work with.

4.4.3. Index Difference Filter

From the stacking operation we also have the index stack containing the index of the image from which the brightest pixel value originated. In these index stacks the satellite streaks are distinguishable from the noise since their index values are identical, whilst a complete satellite track is distinguishable as the index continuously increases or decreases by 1. For these images a kernel operation was developed that checks if neighboring pixels have the same or nearby values. By calculating the total sum difference between the index entry and it's neighboring indices, it is possible to filter out the noise. This is defined by the following expression:

$$F(y,x,s) = \sum_{i=-s}^{s} \sum_{j=-s}^{s} I_{i,j} - I_{y,x} \tag{4.3}$$

Where (y,x) and (i,j) correspond to the position on the images, and s is the size of the filter. For a filter size of $s = 1$ the kernel operation looks as follows:

$$F_{i,j} = \sum \left(\begin{bmatrix} I_{i-1,j-1} & I_{i-1,j} & I_{i-1,j+1} \\ I_{i,j-1} & I_{i,j} & I_{i,j+1} \\ I_{i+1,j-1} & I_{i+1,j} & I_{i+1,j+1} \end{bmatrix} - I_{i,j} \right) \tag{4.4}$$

The boundaries of the new image are calculated by assuming that the entries beyond the original size are equal to zero. Therefore the boundaries will always have a low value in return, which avoids that the borders might remain as a straight feature. Since the satellite tracks are usually only a few pixels wide, using a larger filter size yields worse responses. This is especially evident for faint and discontinuous features as these are drowned out by the noise to a higher degree. After testing several filter-sizes (1,3,5 and 7), the filter-size of 1 (kernel of $3x3$) yields a good response and is further used.

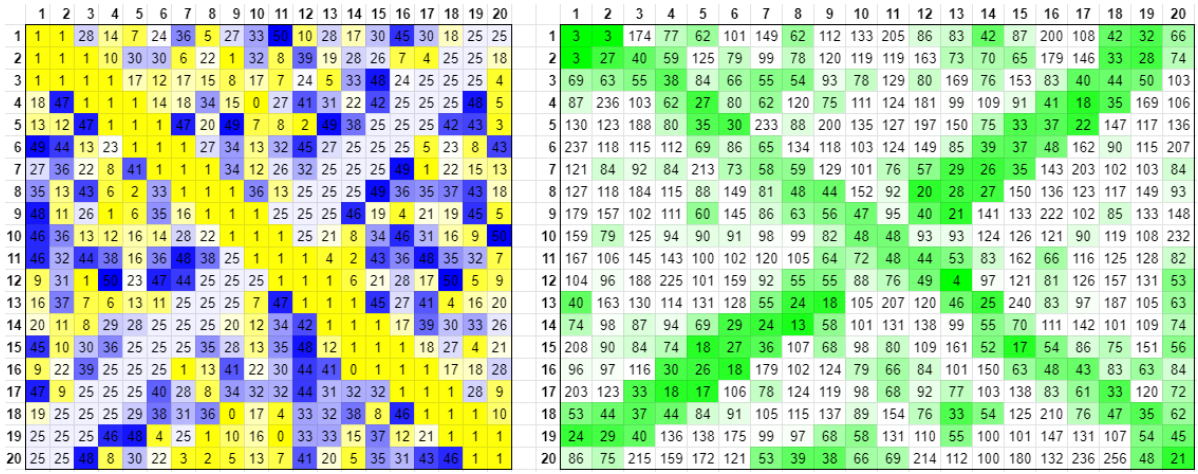


Figure 4.4: Simplified response visualization of the index filter for a kernel size of 3×3 (filter-size = 1). There are two streaks present in the 20×20 frame that run from the corners of the frame with different indices. The other cell values are generated by a uniform random distribution with a range between zero and 50 and thus an expectation of 25. The lower the response from this operation, the better. The feature with index 1 has a higher and therefore worse response than the one with index 25, as it is further away from the expectation.

Larger sizes had the tendency to remove fainter and thinner features as they were often only 2 or 3 pixels wide, meaning that the response would contain some background noise. However, this can easily be changed at a later stage if it turns out that faint or thin features are not useful for endpoint determination. The downside of this filter is that it does not perform that well for slow moving features, as their gradient in the index stack image is very steep. This is another reason why a smaller kernel size is beneficial. However since the main targets are in LEO, fast moving objects are the main use-case, thus the shortcoming of this filter is acceptable. Another shortcoming of this method is that tracks that are around half the index range (25) have a better response than index values near the boundaries of the range $(0, 50)$. This is because the expected background noise level is 25, meaning that an index of 24 has a better response than 49. A good response is a low value since all the cells within the kernel are the same or are close. Features that are near the expectation have better responses from this index filter. This became even more apparent for larger kernel sizes, adding to the reasoning for selecting a smaller kernel size. To show the response, a sample figure is made that contains two diagonal tracks in a 20×20 frame. This is visible in fig. 4.4.

Median filter

An commonly applied filter that is similar to this operation is the median filter [7]. This kernel operation determines the median value from within the kernel, and returns the value to the central pixel. The filter was applied to the index images to test how well noise was removed and two findings were found which suggest it to be inadequate for this application:

- To get a robust approximation of the median value in the kernel, a large size is desirable. similarly to the index filter this blurs out the narrow features, which is counter productive.
- To create a binary image from the median filter response a lot of data was discarded.

After testing a few sizes, the most reliable kernel size was deemed to be 7×7. The next step is to create a binary image from the filtered image. This can be done in two ways:

- By applying the index difference filter that was just described, further removing noise but also removing features.
- By specifying two thresholds that are an integer value away from the expectation of the image (25).

The index difference filter for binary creation removed a lot of data, as there are two steps of noise removal which also remove a lot of features. The second method also reqires us to throw away tracks

that are within the range close to the expectation. After some testing it turned out that the thresholds for a decent response needed to be ± 10 indices away from the mean. This resulted into discarding about 40% of the information in the index images, which is very costly.

It was for those reasons that the median filter was not applied for removing noise in the index stacks. The index difference filter for a kernel size of 3×3 had better responses and the output is also what will be used in the next steps.

4.4.4. Binary Propagation

The three mentioned methods for highlighting the tracks are by themselves useful for the detection method. However, with inspiration from the existing FOTOS1 method we can combine the three products together to create a reliable consensus for the tracks. To do this we first need to transform the three products into binary images, after which we perform an AND filter on the images. To create a binary image we need to define the threshold when pixels are given a binary value of 1. By using the sigma clipped stats from the *Astropy* library [4], we can reliably determine what the background level of the image is. The sigma clipped statistics return the mean and standard deviation of a data subset in which outliers are disregarded (like stars and satellite tracks). For this we use the standard parameter of the function of 5σ. We then set the thresholds such that we create binary images where there is still noise present. such a method ensures that we don't remove faint features straight away. Since there are three different products, we can use the different noise responses to our benefit by multiplying them with each other, eq. (4.6). This should leave a consensus of what are tracks and what is noise. The expectation that since the noise response from the operations is different between the three products, the noisy pixels are omitted whilst the consistent features are kept. There is of course a strong correlation between the input produtcs as they are based on the same input image with the same noise profile, however the operations that are performed result in slightly different binary outputs. The formula for creating a binary image with the sigma clipped threshold (SCT) is defined in eq. (4.5).

$$\text{Output image} = SCT(\text{Input image}, \sigma) \tag{4.5}$$

$$\text{Detect image}_g = SCT(\text{ASBG Sub}_g, 2) * SCT(\text{E-O Difference}_g, 2) * SCT(\text{Index Filtered}_g, 1.5) \tag{4.6}$$

The thresholds chosen for the full range images are 2σ for the images that follow from the value stacks (ASB subtracted image and Even-Odd Difference image), and 1.5σ for the Index Difference Filtered image. As mentioned before the binary images that are created still need to have some leftover noise. For the value stack images it turned out that using a $1-\sigma$ threshold left too much noise, whilst the $3-\sigma$ threshold left too little noise and removed faint tracks from the images. The value of $2-\sigma$ was chosen as a usable middle ground where the response between the two images was also similar. The index difference filtered image was very sensitive to the σ threshold. A threshold of $2-\sigma$ for this image still left similar amounts of noise as the $1-\sigma$ threshold for the other images. To approximate consistency between the three images the $1.5-\sigma$ threshold was chosen. The resulting images from the thresholding can be seen in appendix A, figs. A.11 to A.13.

4.4.5. Moon masking

Besides stars and satellites, the moon is naturally also present in some images. When the moon is sunlit it causes a big bright spot on the image as well as flares that span across the image. Even in cases where the moon is not directly in the frame, it can leave flare features on the images. The bright spot and flares are both features that complicate the detection step as there are clusters of positive pixels. Therefore, we define a mask that is able to cover both the moon and its flares.

We start by calculating the altitude of the moon, and if this altitude is 5 degrees above the horizon, we continue with the masking process. The next steps are described and are also shown in fig. 4.5. The moon's sky position determination is done with the help of the Astropy library [4] using the *get_moon* command. For each stack we can convert the moon's sky position from the function to the x, y pixel position by using the AS as a reference. By specifying a negative margin for the *get_moon* function, we are also able to determine the moon's position beyond the image borders. The moon's position is computed for the start and end time of the stack, as the features tend to drift about fifty pixels in one stack. For the two positions of the moon (1 and 2 in fig. 4.5), we define two lines that intersect the moon

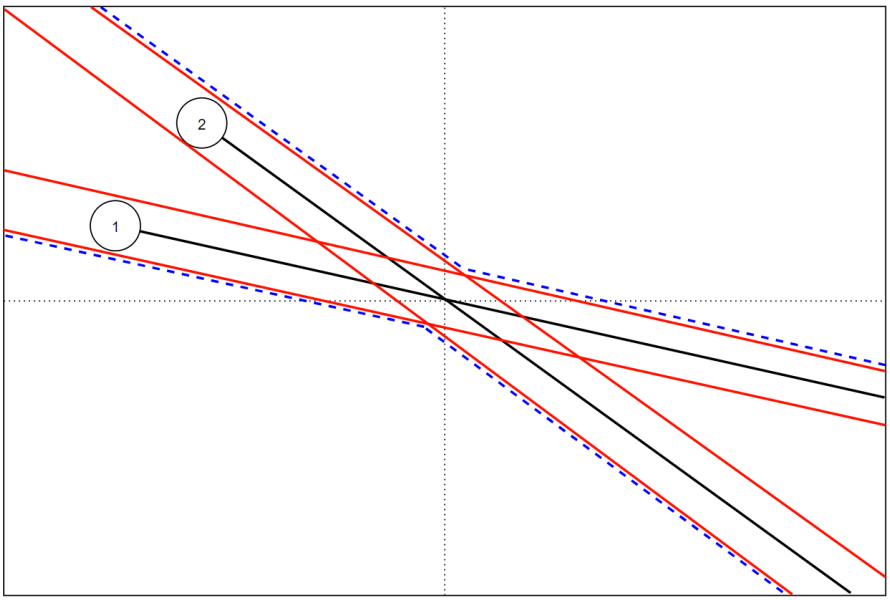


Figure 4.5: Moon masking procedure for removing the moon itself and its lens flares. The region that is within the two dotted blue lines is the moon mask. Note that the size of the moon in the image as well as the change in position over 320 seconds is exaggerated. This figure is for illustration purposes to show the procedure for masking the features.

position and the center of the image (black in fig. 4.5). Along the two respective intersection lines, two offset lines are defined (red in fig. 4.5). The offset selected is 100 pixels (the total width) and covers the expected drift of the features by a factor of two. This offset also covers the majority of the moon in the image and the width of the flares. The outer bounds of these offset lines then serve as the mask borders (dotted blue in fig. 4.5). Whatever is within the moon mask is set to zero in the final detection image, with the risk of also removing some of the satellite tracks. This does not directly mean that we remove the chance of detecting a masked satellite track, as the satellite track might be visible in the next stack where it might not be masked. Besides the mask an erosion operation is applied to remove the remaining noise, this will be further discussed in section 4.4.6.

The current implementation of the moon masking is useful for removing the moon itself and its direct flares, however there are also indirect flares present in some images due to reflections of the instrument. For the southern camera in the La Silla MASCARA data there are some indirect flares that move at the same rate as the moon's direct flares, that cannot be derived from the moon's sky position. The expectation is that some moon reflections are coming from the metal dome of the instrument. Removing these flares is not trivial, neither should it be wanted to do this with image processing. The best solution for removing these indirect flares is by changing the instrument design.

The parameter for the mask offset can easily be updated if it turns out that there are still flares within the detection image or if it is chosen too big. This can iteratively be tuned but requires more testing data.

What is currently not implemented is the possible brightness of the moon in the image. Over time the moon undergoes certain phases where the brightness and size of the moon are different. For MASCARA, only testing data for consecutive days existed, thus it was not possible to implement and validate a possible masking method that accounts for the phase of the moon at that time, as the current implementation would also mask whilst there is new moon (not sunlit moon). This is outside the scope of this thesis and therefore recommended for future research. It solidifies the request for more validation data, possibly consisting of nights with full moon, new moon and one of the quarters. It would be key to changing the mask width parameter according to the expected moon brightness.

4.4.6. Erosion

The erosion operation is done to remove the possible remnants of single noise pixels in the detection image. An erosion operation removes the pixels that contain less than a specified number of neighbors [7]. For normal cases the erosion parameter is set to 1, meaning that isolated pixels are removed. As

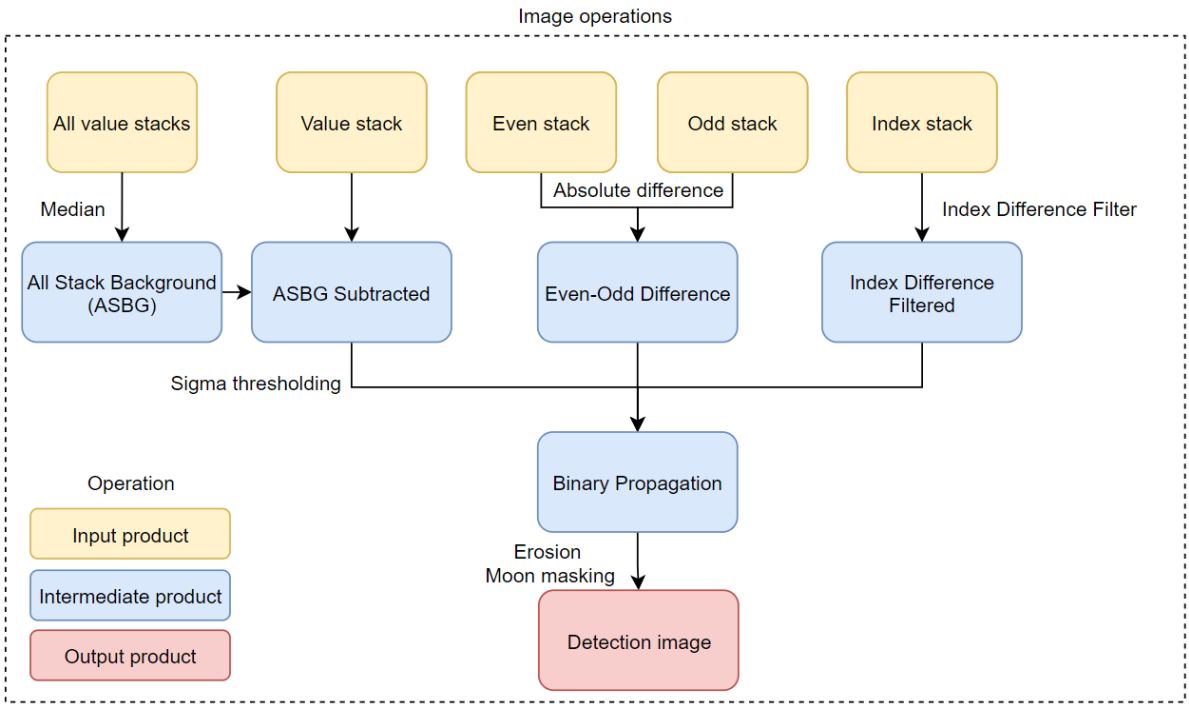


Figure 4.6: Procedure on how the detection images are made.

mentioned in section 4.4.5, the noise level in the image is higher than normal since there is a localized bright spot where the moon is positioned in the frame. Therefore, the erosion threshold is set to a higher value when the moon is present to remove more signals, namely 3.

4.5. Final Products and conclusion

The resulting product from the image processing operations is the detection image, one for each group. The detection images are the binary images that are used for the feature detection methods. The process on how they are made is summarized in fig. 4.6, where they are formatted in red.

Besides the detection images, we also create determination images for each stack. After the tracks have been detected, the determination images are used for endpoint determination and optionally for classification. This is discussed later in chapters 7 and 8.

The detection image can also be combined with the index image to create the Detection Index image. By multiplying the binary detection image with the index stack image, we create an image that contains line-like features with increasing or decreasing index value. A small correction should be made before multiplication by adding 1 to the index image. After multiplication with the detection image, resulting values of zero are ambiguous, as zero can either be due to the index image or due to the detection image.

Note that we did not apply a correction for the PSF of the cameras, as this helps in highlighting the features due to the increased width. Since the features can become curves over longer sections the feature width can help in extending the distance of the feature that can be detected as straight.

Also the selected test data set (2020-01-03, La Silla) is nearly complete and had clear skies all night. Only during the beginning of the night there was a bright moon present, which needed to be accounted for. The effects of cloud formation and the presence of different phases of the moon were not tested on this night.

More test data is needed such that we can cover these situations. For instance data from winter time in Chile is can be used to look for higher objects and extra validation of the data reduction simulation. The effects of missing data in the stacks also needs to be still be tested for, however for this data set the last image in every sequence was missing already. In terms of stack creation this should not be a problem, however the subtraction routine should be adapted by subtracting nearby frames. Also the presence of gaps within stacks or the whole night could lead to different output products. This can

however be tested by selecting a sub-sample of the current testdata. This should still be investigated before the pipeline becomes operational.

5

Feature analysis

This chapter describes the analysis of the satellite tracks in the images. The results from this analysis serve as an input for the feature detection chapter that comes afterwards. As mentioned before, a track consists of multiple subsequent streaks or streaklets. So with the stacking methods we are looking for satellite tracks, whereas the existing FOTOS1 method was looking for satellite streaks. For a single streak there are two data points derivable, the start and the endpoint. For a stack of 50 images, there are (ideally) 51 data points since the overlapping ones are (in theory) equal. For single difference images the assumption that the features are straight lines was sufficient, however, within stacks the features become longer and can also become curved. The scale of the deflection from straight and the source(s) of it are investigated.

To determine how the satellites are perceived by the observer it should be clear which effects play a role. If we define the celestial background as stationary, the observer and the satellites both move within this frame. This is shown in fig. 5.1.

Since we use the AS to overlap the celestial backgrounds onto each other, the relative movement and the observation angle is what defines the satellite features on the image. The movement of the observer with respect to the celestial background is known, however the movement of the satellites is possibly unknown. The ambiguity of how the satellites are moving on the frame plays a role in the feature shape. An object might be passing by close and fast, but this could be due to a circular or an highly eccentric orbit. These dynamics are further discussed in section 5.2.

5.1. Reference Satellite Positions

For determining the performance of the developed methods, the position of known satellites is needed as a reference. By using the Simplified General Perturbations model the position of cataloged satellites can be determined [30]. Per object the method returns the position of the object in the True Equator, Mean Equinox (TEME) frame. This TEME position is then transformed into the equatorial coordinate system such that we can retrieve the right ascension and declination of the object (celestial coordinates). This transformation routine was done using transformations from *PyEphem* [24]. These positions in right ascension and declination are then transformed to the pixel position on the frame with the AS. The AS uses the Kharchenko catalog for determining the observed section of the celestial sphere, this is discussed in [14]. There are thus two sources of errors in this positioning of the satellites on the frame; the reference AS and the position from the TLE. The error of the AS is estimated to be 0.1 pixels for the majority of observable stars in the frame.

The catalog of Two Line Elements (TLEs) that is used is taken from Space-track[1] and matches the date of the testdata. A single TLE contains information of the satellite object that defines its orbit, age, identifier and several other parameters [31]. For each timestamp within a stack we determine the satellite's position and check if it is in the current frame. The timestamps are defined as the endpoints of the tracks, and are set as the start and end of each exposure. Since the exposure time is 6.4 the endpoints are defined as the image JD ± 3.2 seconds. The exact exposure time and JD of each frame is extracted from the header of the fits file as was mentioned in chapter 2. Since subsequent streaks

[1] https://www.space-track.org/, the catalog of 2020-01-03

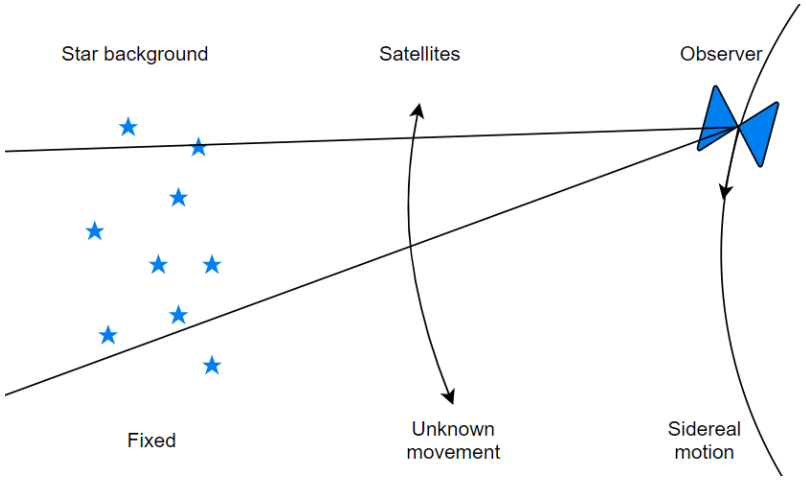


Figure 5.1: Visualization of how the features on the images are perceived. The stars are considered stationary, the satellites move in a possible unknown orbit and the observer (blue hourglass) moves at the rate of earth's rotation. The triangular shape gives an idea for an off-center observation of satellites. The top defines the features, the bottom defines the dynamics of the feature.

share their start and endpoints, the average timestamp of the overlapping points are used. The potential inaccuracy that is caused by this is discussed in section 5.3.

5.1.1. Accurate TLE sets

The positional accuracy was studied in [6], and for LEO objects the positional error was at most 1 kilometer (see their Fig. 1). From the study it is evident that there is a negative correlation between the object altitude and the positional error (lower altitude equals greater error). A quick estimate for the worst case is that for an observational distance of 500 kilometers the pixel discretisation is 0.17 kilometers (0.02 degrees per pixel). This then translates to about a positional error on the frame of approximately 6 pixels. Note that this is the worst case for objects with low altitude. It is therefore advised that for checking the performance of the algorithms for objects in LEO, a comparison needs to be done with objects at the upper range of the LEO domain (between 1,000 and 2,000 kilometers). This avoids errors due to wrong reference positions.

Besides the method itself, another possible source of inaccuracy is from the TLE catalog itself. The TLE format does not have a gauge on how accurate the estimate of the orbit is. A recent epoch (update of the TLE) also does not guarantee the validity of the orbit estimate. The majority of the objects in the 2020-01-03 are 6 hours to 3 days old, meaning that they are all quite recent. However the Simplified General Perturbations propagator (SGP4) model is limited in estimating future positions over longer timespans.

The expectation is that whenever there is an inaccuracy of an object's reference position, it would be difficult to be matched to a detection and would therefore not be included in the endpoint determination. We have to make sure that the method for matching a satellite to a detection is very strict such that we avoid positional errors of inaccurate objects.

The groups of satellites that are used for comparing the performance of the detections and endpoint determinations are stated below:

1. Near circular objects in the upper range of the LEO domain. Where the satellite altitude needs to stay within the boundaries of $[1,000 - 2,000]$ kilometers.
2. Near circular objects beyond the LEO domain. These are objects with an eccentricity below 0.1 and an altitude that is always higher than $2,000$ kilometers.
3. Eccentric rocket bodies with low perigee and high apogee. These are objects with their lowest point (perigee) below $1,000$ kilometers and their highest point (apogee) beyond $2,000$ kilometers.

After an object is detected it will be checked if it fits into one of the three groups. If it does not it is placed in group 0. The detection performance and the endpoint determination accuracy can then be returned into more sensible groups of objects, as a total number might not give good insight.

The number of endpoints per satellite in one stack depends on the number of images that are within the stack, and if the satellite is in view of the camera the whole time. For FOTOS1 each difference image contained two endpoints per streaklet. This resulted into often having two estimates for the same timestamp. For the new method there are no duplicate endpoints since the streaks overlap within the track.

5.2. Satellite Track Shape

The shape of the tracks on the images are not expected to be perfect straight lines as this only happens under special circumstances. From initial investigations of the data points from the catalog, it was clear that for multi exposure tracks the features are not linear but rather curves. For the detection of the features it helps if the features are straight, as this simplifies what needs to be searched for. If for instance, curvature needs to be assessed, we increase our search space with extra parameters. Therefore it is of interest to know what the approximate length will be, where the features can be detected as straight, and at which length this assumption starts to become untrue.

The curvature of a feature can be caused by one or multiple effects. Each effect will be described and discussed, after which we determine the expected length, width and curvature of the features.

Satellite Orbit

The movement of satellites within the images is ambiguous as there is little to no information on the observed distance of the object (only if we have a priori data). The length of individual streaks can be used for determining the velocity, however, this does not directly translate to an altitude as it might be an eccentric object instead of a circular one. For circular orbiting objects the features will appear straighter as the altitude increases. This is because the observed movement over time with respect to the star background is less, meaning that the object moves in a straighter segment.

Sphere projection

The projection error is introduced when a spherical object is mapped onto a flat plane. The rectilinear projection causes a loss of information of depth, making everything appear at the same distance. Especially when objects are moving across the frame this causes changes in apparent movement. This is also the case when objects move in the direction of the observation direction, as this movement does not translate well to the flat frame. This relative movement error in this application becomes more apparent when the satellite vector (earth center - satellite) and the observer vector (earth center - observer) are at an angle, this can also be defined as the observation angle.

Lens aberration

The deviation that occurs between the perfect projection and the perceived projection is called distortion and is an optical abberation. The error is introduced due to the shape of the lenses and it's possible imperfections. The distortions are often centered around the optical axis and symmetric. For wide angle lenses barrel distortion is often present. This is a common minor distortion that can optionally be corrected for with information of the optics. However as will be shown later this error is minor and is not detrimental to detection performance in comparison to other errors.

Observation Angle

The observation angle of the cardinal direction cameras causes a significant distortion in terms of perspective. For satellites there is only one perfect condition where the features are perceived as straight- when the pass is directly overhead in the zenith camera. When the camera is pointing at a slightly lower elevation two things happen; the rate at which stars drift is different and the perceived motion of satellites changes.

5.2.1. Length

The length of the features is dependent on two things; the orbit of the satellite and the observation angle. The satellite orbit itself determines how quickly the object moves over the observer (velocity), and the observation angle determines how that movement translates to a 2D image.

The movement of the satellite is determined by the orbit that it is in, which is defined by the six Kepler elements. The majority of the operational satellites have (near) circular orbits, meaning they have little to none eccentricity (e) and thus their altitude is constant. The altitude can be derived from

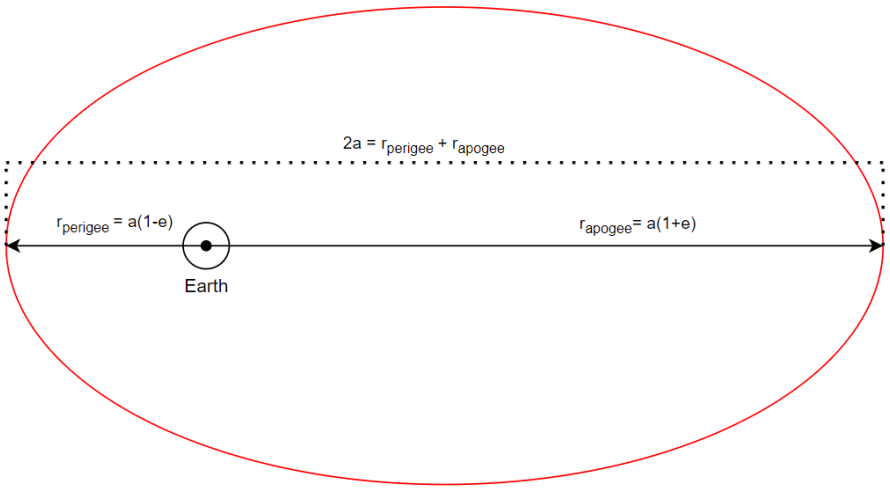


Figure 5.2: Visualization of an eccentric orbit and how it defines the range where it operates. The relations for the apogee and perigee are defined in the semi major axis and eccentricity.

subtracting the Earth radius from the semi-major axis (a)- the first Kepler element. However, for larger semi-major axes the number of objects that have eccentric orbits increases. The objects with eccentric orbits are often rocket bodies that transfer the satellite from a lower orbit to their operational orbit. After the insertion of the satellite in the operational orbit, the rocket body is often left in their transfer orbit. An example of an eccentric orbit is shown in fig. 5.2.

For circular orbits the velocity of the object is consistent, however, for eccentric orbits the velocity depends on the position along the orbit. The object moves faster at perigee and slower at apogee due to the gravitational potential at the respective positions. The velocity therefore varies during one revolution of an eccentric orbit, which is difficult to directly observe in the images without a priori knowledge.

To show the effect of the observation angle of the satellite we use a circular orbiting object as a reference, and estimate the traveled distance (and thus indirectly feature length) for two different viewing angles. For circular orbits the semi-major axis relates to the orbital period with the next equation:

$$T = 2\pi \sqrt{\frac{a^3}{\mu_{earth}}} \tag{5.1}$$

For a single stack the time passed is 320 seconds, with this we can express a relation between the semi-major axis and the fraction of the orbit that has passed in one stack. The arc that has passed in 320 seconds in a circular orbit is expressed in anomaly (θ in degrees). However the arc that has passed in the orbit is not equal to the arc that is observed by the observer (local coordinate system). Intuitively this makes sense because the the satellites orbit the Earth and the observer is not at the center of the Earth (but at the radius). The perspective that the observer has of the satellite is thus different and is shown in fig. 5.3.

For different altitudes of circular orbits, we can compute what the perceived arc length on the images are and thus the length in pixels. By computing the change in angular position (δ) we can directly translate the angle to pixels with the 0.02 ratio mentioned in chapter 2. The change in elevation is estimated for the most ideal case (zenith, directly overhead), and the most extreme case (zero elevation, at the horizon). The most extreme case is a very safe estimate since the minimum elevation of MASCARA is about 20 degrees.

Since we express the δ angle against two perpendicular lines (horizontal and vertical), only one angle has to be determined. The angle γ is determined by evaluating the orbital period of the satellite and the anomaly that has passed in one stack (320 seconds). Then the distances between the observer and satellite are estimated, where d_1 is a trivial estimate as it is either the orbital altitude (zenith) or the hypotenuse between the semi-major axis and Earth radius. The parameter d_2 follows from the cosine rule as two members and the angle between them is known. For more clarity the expression is given to calculate d_2:

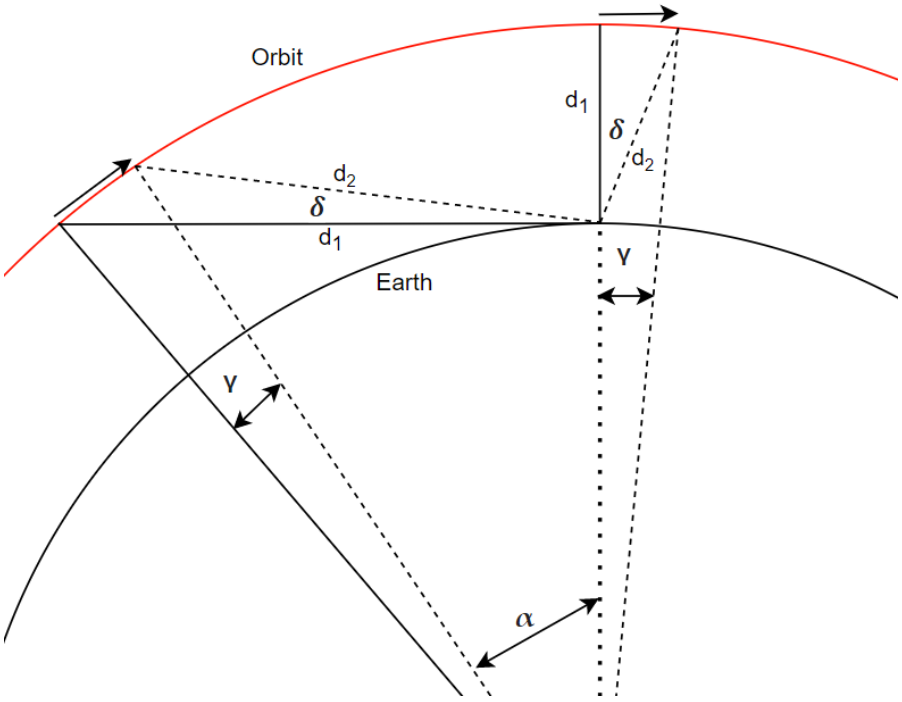


Figure 5.3: The geometry between the observer and the satellite orbit in 2D. In red we can see an orbit at approximately 2,000 kilometers altitude, which is approximately shown to scale with the earth. The observer is located at the top, where two looking directions are defined; straight overhead and on the horizon. The angles γ for both situations are equal, but the angles δ are different due to depth perception. The distances d_1 and d_2 represent the observation distance at the two moments in time, these are needed for calculation.

$$d_2 = \sqrt{a_{sat}^2 + R_{earth}^2 - 2a_{sat}R_{earth}\cos\alpha} \tag{5.2}$$

The parameter a_{sat} is the semi-major axis of the satellite and relates to the altitude by subtracting the earth radius from it. The angle α is the one between the line segment d_2 and zenith, for the overhead situation this is equal to the angle γ. The change in elevation δ is then estimated with the sine rule as two members and one angle in the triangle are known. This is given as:

$$\delta = sin^{-1}\left(\frac{a}{d_2}\right)sin(\gamma) \tag{5.3}$$

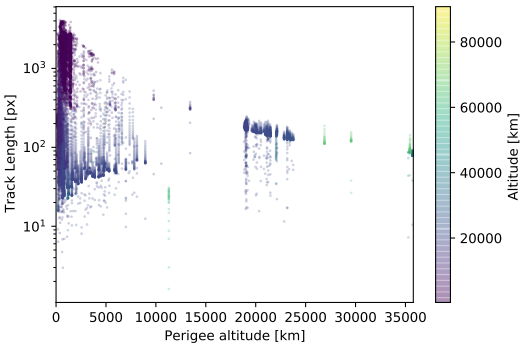
For three altitudes both the zenith and horizon situations are estimated for, where we convert the expected change in angular position to feature length on the image plane.

Table 5.1: Expected feature length for circular objects and neglecting earth rotation. Calculation performed for multiple altitudes and observation angles. The altitudes are chosen to show the feature length range in the LEO domain and the upper extent of the space domain (Geostationary Orbit, GEO). The orbital period (T) is estimated with eq. (5.1) and compared against the time passed of one stack. The percentage of the orbit passed in the time of one stack is expressed in a percentage (obs.) and in anomaly ($\Delta\theta$).

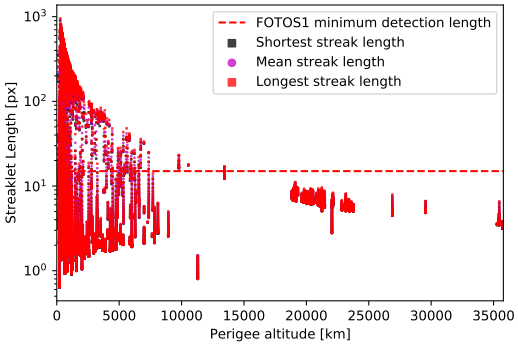
Object	Type	h [km]	T [s]	Obs. [%]	$\Delta\theta$ [deg]	d_2 [km]	δ [deg]	Track [px]
LEO	Horizon	400	5554	5.76%	20.74	414.93	74.10	3704.82
LEO	Zenith	400	5554	5.76%	20.74	2401.01	89.06	4453.01
LEO	Horizon	2000	7632	4.19%	15.09	3778.01	18.43	921.69
LEO	Zenith	2000	7632	4.19%	15.09	2772.62	51.90	2594.83
GEO	Horizon	35786	86164	0.37%	1.34	41530.12	1.34	66.97
GEO	Zenith	35786	86164	0.37%	1.34	35788.05	1.58	78.76

What can be deduced from the results presented in table 5.1 is that the track lengths within the LEO domain are able to span almost the entire frame (diagonal is 4817 pixels). And it is also evident that the features are perceived shorter when looking at a low elevation due to the projection error. This is due to that these objects also move towards or away from the observer, and movement in depth can not be picked up in a 2D image.

For checking the effect that the eccentricity has on the feature length we can plot the object track length against the perigee altitude of the object. The data is taken from all the satellite passages in the central camera in the La Silla MASCARA instrument for the night of 2020-01-03. The central camera is selected since it is influenced the least by projection error. Both the length of the satellite track and FOTOS1 streaks are shown, where the length is the absolute distance between the outer most points. The results are shown in fig. 5.4.



(a) Plotting the track length against the perigee altitude of the satellites from the TLE catalog visible in the La Silla Center camera. The color index defines the altitude of the object.



(b) Plotting the streaklet length against the perigee altitude of the satellites from the TLE catalog visible in the La Silla Center camera. There exists a limit of detection performance of the existing method as it can not detect objects beyond 15,000 kilometers perigee altitude.

Figure 5.4: Comparison of track length and streaklet length of the new method and the existing FOTOS1 routine. It shows the increase in potential of detectable features.

fig. 5.4 illustrates how the track length decreases asymptotically as the perigee altitude increases. When we compare this to the existing FOTOS1 method we can also conclude that the existing FOTOS1 routine is unable to detect any object beyond $15,000$ kilometers perigee altitude as the method was only able to detect features with a line length of at least 15 pixels long.

Since the MASCARA and bRing instruments do not track the stars, they move with rotation of the Earth. As mentioned before, the exposure time of the cameras is set to ensure that stars remain within a single pixel on the frame. With the AS the images in one stack are moved such that the stars overlap. The movement that this causes is apparent by looking at fig. A.1, as the hot pixels show the movement of the frame with respect to the stars. The features that the hot pixels leave is later discussed and accounted for in chapter 6. This position shifting due to star alignment also influences the satellite features as it causes the streaks to become slightly misaligned. The effect is however dependent on the relative movement of the satellites against the star movement and is in the order of 1 or 2 pixels:

- If the satellite moves along with the stars, the satellite streaks appear shorter or longer depending on the movement direction.
- If they move perpendicularly, the streaks are of normal length but the start and endpoints of streaks no longer align perfectly.

The worst case scenario (perpendicular to star movement) is shown in fig. 5.5. Perpendicular movement is undesirable since it causes the streaks to be misaligned, whereas with the movement that is in-line with the star movement only results in gaps and overlaps. This is already an issue due to the feature width, which is later discussed in section 5.2.2.

The misalignment issue is negligible in terms of detection as the features are wide enough to still be aligned by some margin. However, it can influence the endpoints as they are expected to be somewhere in between the two streaks. This is something that should be taken into account in the endpoint

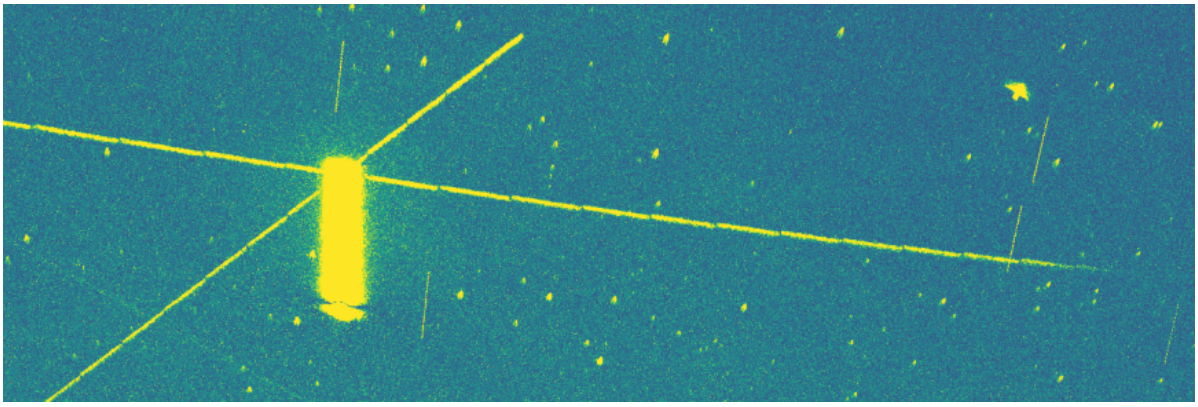


Figure 5.5: The alignment error of streaks within a single satellite track due to background aligning. The shorter and thinner features that are perpendicular to the satellite track are the remnant hot pixels that are shifted due to star alignment.

determination methods. Moreover, since the subtraction routine is from subsequent images, the streaklets overlap a few pixels due to the PSF. This also causes a small gap between individual streaks within a track (about 3 pixels). This is also an issue for the endpoint determination, as the crossover point between the streaks is where the endpoint exists.

The positioning error that the AS introduces is at most a single pixel per stack, as that is the accuracy of the AS itself [29]. When comparing this to the effect of the satellite orbit it is minor for LEO objects but can become more influential for higher objects. However, since the main goal of this research is for LEO objects, the effect of stacking relative to the stars does not introduce an error substantial enough to be compensated for.

5.2.2. Width

The brightness of a satellite feature on the image is expressed in magnitude, which is expressed relative to the sun's apparent magnitude.

$$\mathrm{mag}_{sc} = \mathrm{mag}_{\mathrm{sun}} - 2.5 \log_{10}\left(\frac{I_{\mathrm{sc}}}{I_{\mathrm{sun}}}\right) \tag{5.4}$$

Where the I denotes the intensity or luminosity of the object. For a spherical Lambertian (matte surface) object in space reflecting sunlight, the brightness depends on a few factors- these are formatted in the next equation.

$$I_{\mathrm{sc}} = \frac{I_{\mathrm{Sun}}}{d_{\mathrm{sc,obs}}} \frac{2}{3} \frac{C_d}{\pi} r^2 (\sin\alpha + (\pi - \alpha)\cos\alpha) \tag{5.5}$$

Equation (5.5) relates the factors of the spacecraft (sc) with respect to the geometry between the sun, spacecraft and the observer. The triangle that is made by the three objects has a phase angle α, which was previously mentioned in chapter 3 (fig. 3.14). The parameters of the spacecraft are the reflection coefficient (C_d) and the radius of the spacecraft (r). The brightness of the sun (I_{sun}) and the distance between the spacecraft and the observer ($d_{sc,obs}$) are the other parameters in the equation. Equation (5.5) does assume there is no clouds or turbulence between the spacecraft and the observer.

As mentioned in chapter 3 the phase angle plays a key role in the perceived brightness of the objects, but also the reflectivity of the object. Another important factor is that the magnitude is defined as the cumulative flux that is observed. This means that the light that is observed in one track is distributed across the whole track length. Therefore, a fair comparison between satellite magnitude and the targeted observational magnitude of MASCARA is not directly possible. The MASCARA instrument is optimized for the visual magnitude range between four and eight [28], but can also observe much fainter objects. In a study from the European Southern Observatory (ESO) [8] a table was formatted with satellite constellations and their observed visual magnitude. The magnitudes varied between 3.2 and 8.7 dependent on the phase angle, altitude etc. For a conservative estimate we can assume a single streak length of 100 pixels, this, in terms of the magnitude, means an adjustment of $\mathrm{mag} + 5$. MASCARA is therefore able to observe certain constellations, however it does depend on the geometry

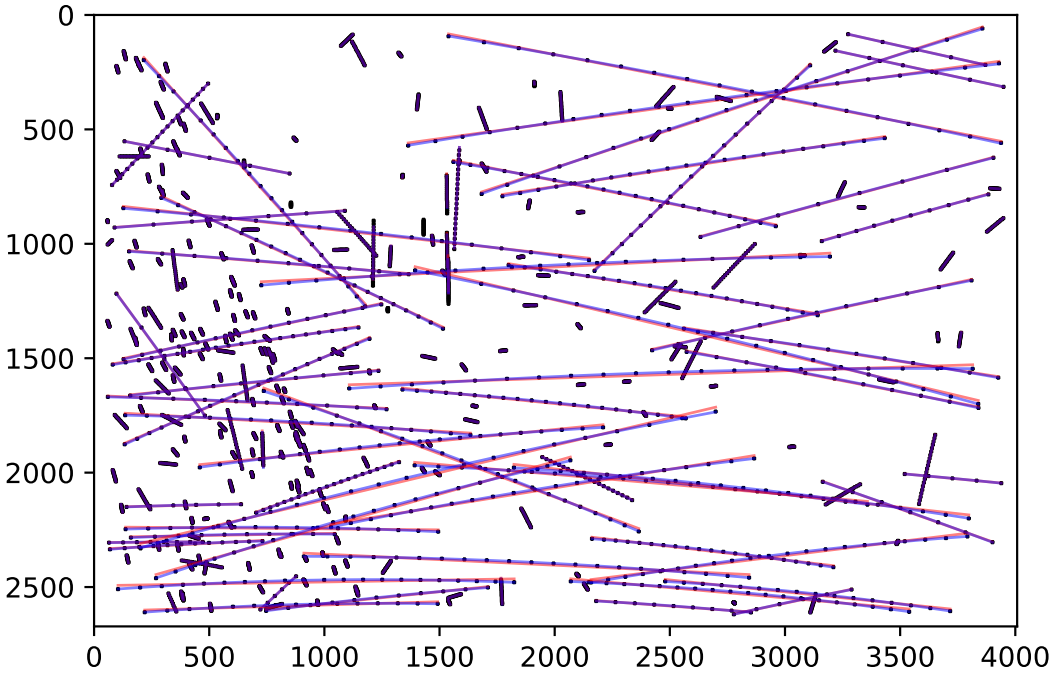


Figure 5.6: A small portion (1%) of what a single camera (East) of the MASCARA instrument sees during one night of observation. The black dots represent the satellite position on the frame according to the TLE catalog. The two lines that are fitted to the data points are linear (red) and polynomial (blue).

of the observation. Much fainter objects, for instance in GEO, are less likely to be detected as they become fainter due to the large distance between observer and spacecraft. However, since they don't move as fast, all their light is binned into less pixels making them appear brighter again.

Bright point sources in images are usually smeared out over a larger area due to the point spread function (PSF) of the instrument. The PSF can translate the effect that the optics have on a perfect point source. If the PSF is well known it can be used to deconvolute the image with it to create a sharper image without smearing. For the MASCARA instrument the PSF function is known to be a few pixels wide, where the outer extent of the PSF is just within 10 pixels in diameter. The function of the PSF is not universal per camera, but it does depend on the position of the frame. For MASCARA this is displayed in Figure 7 of [28]. In the center of the frame the PSF is more evenly distributed whilst towards the edges and corners it becomes more elongated. Because of the PSF the satellite tracks appear wider than a single pixel, mostly 5 pixels wide with an extent up to 10 pixels. The extent of the PSF is often expressed in Full Width at Half Maximum (FWHM), but no exact figure for this was found in literature of both bRing or MASCARA. In [27] the FWHM of the PSF is mentioned to be larger than 2 pixels, but the exact size is not given. The FWHM also does not give a limit to the outer extent of the PSF, as it only gives the width of the region where most of the signal goes to.

5.2.3. Curvature

To estimate the curvature of the features in the images we make use of the TLE catalog and determine the position of each satellite, during the whole night, for each camera. All the positions of each object that are within one stack are then analyzed to see how well they fit a straight line and a second order polynomial. The fits are done by using a least squares regressor to the available data points. The absolute difference between the linear fit and the polynomial fit gives an indication of how far the feature deviates from a line approximation. An example from a portion of real testdata is shown in fig. 5.6.

It can be seen that for the longer features on the frame the curvature is more present. Due to the fitting with least squares the greatest deviation occurs near the boundaries of the feature as well as in

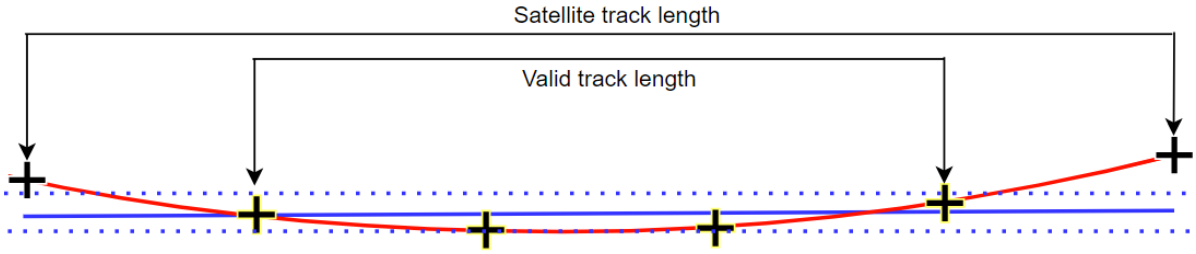
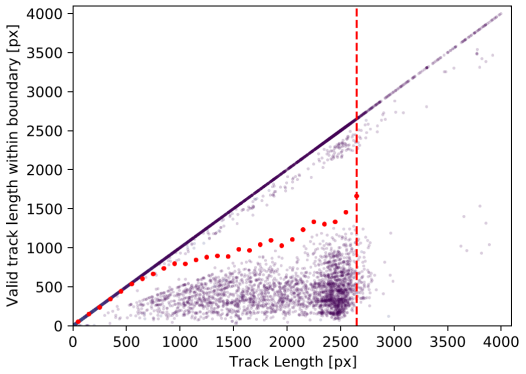


Figure 5.7: Definition of the valid track length compared to the total track length. For a given satellite track there are a number of data points available (crosses). They are fitted with a linear and polynomial fit, where the difference between the two is the error at each data point (where linear is the reference). The margins define the hypothetical width of the features, where the longest segment between the margins is equal to the valid track length (marked yellow) of the available track length. For every satellite position the difference between the linear and quadratic fits are checked, and if this difference is greater than the margin it is removed from the valid points.

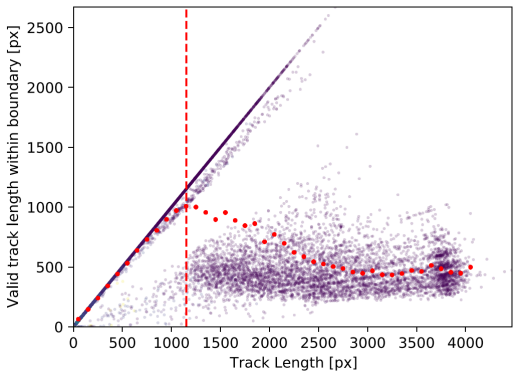
the middle. For determining how curved the features are, we use the difference between the two fits to define two metrics.

- The Mean Absolute Error (MAE) between the linear and polynomial fit, this gives an indication as to how well the satellite track can be perceived as a straight line. This is determined per unique satellite track.
- The Valid Track Length (VTL) is determined by checking the fits of the track and seeing for what length the polynomial fit falls within the boundaries of the PSF. This provides a metric up to which length the tracks can be detected as straight. This is also determined per unique track. A visualization of how the VTL is determined is shown in fig. 5.7.

To show the limitation of when features can be detected as straight lines we sub-sample the valid track length for groups of increasing track lengths with 50 pixel increments, resulting in the mean valid track length for a sub-sampled group of track lengths. These results for two different cameras are shown in fig. 5.8. When we compare the responses for the two looking directions it is evident that the center camera contains straighter features. The cameras that observes at a lower elevation angle have features with more curvature since the movement on the frame over time is less consistent. The responses for the other cardinal directions (north, west, south) have comparable responses to that of the east camera. The differences between the cardinal directions depends on the type of objects that are within each frame. For instance the northern camera observes more objects at geostationary altitudes compared to the southern camera. Also the movement of the stars for each camera is slightly different, and this causes the alignment of the stacks to also be different.



(a) Valid track length against track length for the center camera.



(b) Valid track length against track length for the east camera.

Figure 5.8: Plotting the available track length against the valid track length, the red dots represent the averages per groups of 50 pixels. The dashed line specifies the length limit that is chosen.

To determine the maximum detectable feature length per camera we use the sub-sampled mean VTL, and return the track length for which the VTL response was highest. For the center camera this

is limited to the frame height (2672 pixels) as the majority of features are shorter than this length. For the center camera the largest VTL response occurs when the track length is longest, as the projection error of the center camera is less significant. For the cardinal directions, flattening occurs between 850 and 1150 pixels. The results are formatted in table 5.2.

Table 5.2: The valid track length drop-off and the mean of the average fit error for linear sections for each respective camera. The VTL drop-off is later used for the maximum line length that is searched for.

Camera direction	VTL drop-off [px]	Maximum MAE [px]
Center	2650	15.0
North	850	35.9
East	1150	16.5
South	950	29.2
West	1150	17.9

The two metrics per camera are formatted in table 5.2 and later used in the track detection stage to optimize the detection method for the expected features. As a consequence,the detection method becomes highly adaptable when the observation strategy changes. The valid track length determines for what type of lines need to be looked for, and the maximum MAE is used for isolating the detected tracks to include the possible curvature of the detected feature.

Besides the viewing angle of the camera also other sources for the curvature were investigated. It is clear that the optical distortion itself does not cause a major error as the central camera still has near straight valid track lengths for larger track lengths (fig. 5.8).

The eccentricity of the objects in view also did not show a clear cause for the VTL. Most of the curvature is due to the projection of the satellite orbits and not the satellite orbits themselves.

Polynomial order of the track shape

Finally, we can motivate by what order polynomial the features need to be approximated by. By using the catalog positions on the frame, we perform least squares fit up to fourth order polynomials and check the mean absolute error of the fit. By calculating the mean and standard deviation over all the objects that are fitted (thus the mean of all mean absolute errors), we can get an idea on how well the fit can approximate the features. The results are shown in table 5.3 and the error distribution is shown in fig. 5.9.

Table 5.3: The overall mean error of the mean absolute error for polynomial fits of the Simplified General Perturbations propagator (SGP4) sky-positions. Orders up to four are investigated as higher order methods did not yield a significant increase anymore. SD = Standard Deviation. The decrease is calculated as a percentage of the previous total.

Camera direction	Linear		Quadratic		Cubic		Quartic	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Center	4.259	21.073	1.432	17.009	1.132	16.332	0.912	16.010
North	3.347	22.296	1.545	19.279	1.235	18.479	1.104	18.042
East	2.163	7.639	0.267	5.400	0.172	5.198	0.128	5.106
South	8.309	29.325	3.087	25.023	2.295	24.051	1.970	23.512
West	2.168	7.428	0.214	5.526	0.130	5.473	0.093	5.416
Average	4.049	17.552	1.309	14.447	0.993	13.907	0.841	13.617
Decrease	-	-	67.68%	17.69%	24.14%	3.74%	15.25%	2.08%

Table 5.3 shows that the linear approximation is unable to fit the features correctly. When applying the second order term in the fit, the error of all looking directions drop significantly. It is trivial that increasing the order of the fit will decrease the overall error of the fit, since the order equals the number of data points it creates an exact representation. When looking at the average decrease in the mean (bottom row), it decreases substantially for higher orders. However, the standard deviation does not decrease significantly anymore beyond the second order (quadratic). It indicates that there are certain features that prove difficult to fit by a polynomial. Because the standard deviation does not decrease much beyond the quadratic fit, it is assumed that the second order polynomial is adequate to model the majority of features without introducing too many parameters. Also knowing that the feature width

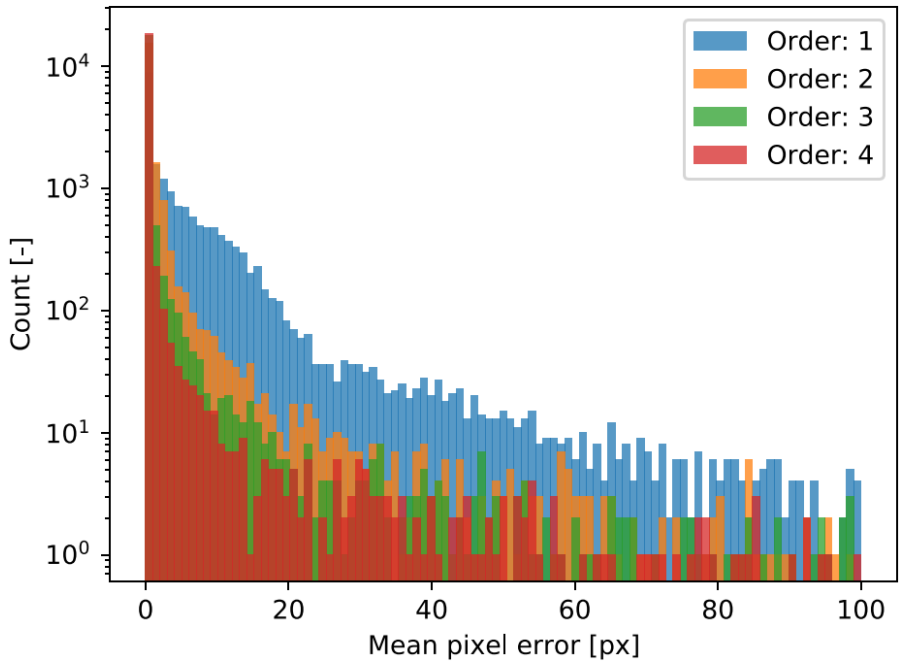


Figure 5.9: Histogram of the mean absolute error of the polynomial least squares fits of the SGP4 points on the image frame. This is the result for the Southern camera, the one with the largest standard deviation in table 5.3.

is often a few pixels wide, supports the application of quadratic functions as the mean error is in the range of the expected track width.

Also in fig. 5.9 it is clear that there are large errors for all orders of polynomials, and that the largest decrease happens when going from linear to quadratic.

5.2.4. Endpoint Spacing

Besides the difference in overall track length, also the streaklet lengths within the track changes when it moves across the frame. This is evident through visual assessment of the even and odd images in appendix A, where the length of streaklets clearly vary. The level of which these endpoints vary highly depends on the observation angle and thus the projection of the motion. Possibly the effects of eccentric movement of satellites can be picked up, but the effects are expected to be minimal compared to the projection effect.

5.3. Exposure Length and start-end differences

The reference time for the exposures according to the header of the images is 6.382525 seconds. However, the exact exposure time is not always equal to the reference that is set. By analyzing the header data of one whole night of observations and comparing the measured exposure time to the reference exposure time, we can estimate if the error has an indirect effect on the sky position of the satellite. A histogram is made of the error in exposure time with respect to the reference, where the respective cameras are also separated from each other.

The error is at most 2.5 milliseconds, percentage-wise this is around 0.04% of one exposure. It demonstrates that the positional error due to exposure time error is negligible, as 0.04% of any feature length is well below 1 pixel in size.

5.4. Light-time correction

For the objects in view, the light that comes from the objects has a certain travel time. This gives an inherent uncertainty in the position of the satellite on the image, as what is perceived as the satellite's position, possibly needs to be corrected for due to light travel time (LTT). The LTT is computed with the

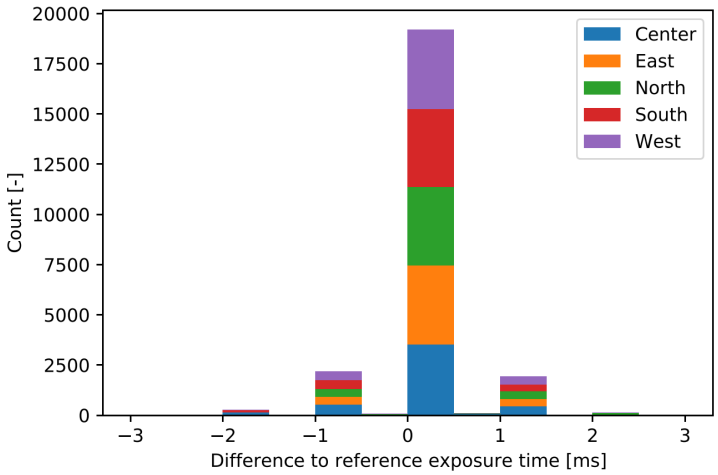


Figure 5.10: Stacked histogram showing the exposure length error in milliseconds with respect to the reference. The data is from one night of observations (03-01-2020) thus there were approximately $4,500$ images for each camera. The majority of the exposures are within one millisecond.

following equation:

$$t = \frac{s}{c} \tag{5.6}$$

Where $c = 299792458$ is the speed of light in meters per second, t is the LTT in seconds and s is the distance in meters. For objects that pass exactly overhead, the altitude of the satellite is equal to the distance between observer and satellite. For objects at the horizon, the observational distance is one of the sides in a right triangle. Since the extent of elevation of MASCARA is at most about 10 degrees, using the horizon is a conservative estimate.

Table 5.4: LTT estimates for different conditions. The semi-major axis (SMA) equals the height + the radius of the earth and the Exposure percentage is the fraction of one single exposure (6.4 seconds).

Type	Height [km]	SMA [km]	Elevation [deg]	Distance [km]	LTT [s]	Exp. [%]
LEO	400	6,778.136	90	400	0.0013343	0.02 %
LEO	400	6,778.136	0	2,294.016	0.0076520	0.12 %
LEO	2,000	8,378.136	90	2,000	0.0066713	0.10%
LEO	2,000	8,378.136	0	5,432.545	0.0181210	0.28%
GEO	35,786	42,164.136	90	35,786	0.1193692	1.87%
GEO	35,786	42,164.136	0	41,678.936	0.1390260	2.17%

Table 5.4 shows that the effect of the LTT can be neglected as it only causes a minor positional error. Comparing these results with those of table 5.1, it is clear that for a single exposure (streaklet) the error due to LTT is well below 1 pixel.

5.5. Classification

Besides the satellite features that are to be detected, there are also other features in the images that can cause false positive detections. From evaluation the most common source of line-like features are the following:

- Airplanes, leaving long features with blinking behavior.
- Bright moon, large overexposed point source in the frame.
- Lens flares, from a bright source like the moon.
- Meteorites, possible entry of space object that burns up (partially) in the atmosphere.

The features that are present because of a bright moon are masked out (section 4.4.5). However, as discussed in previous chapters, not all flares were resolved successfully. There exist some indirect flares from the metal dome of MASCARA, which cause flares that are not in-line with the position of the moon. For future use of the software it is recommended that the instrument is free of possible flare sources. This can be achieved by introducing some hardware changes. Especially the indirect reflections in the southern camera from the dome can be eliminated by making the surfaces less reflective (blackening them).

For distinguishing the other sources from satellite streaks use can be made of the brightness information of the object in the image. It is known that airplanes employ a blinking pattern for their identification, so this can be extracted from the feature to classify them as one. The brightness behavior of satellites can vary, as satellites that are still in operation have attitude control. When a satellite no longer has attitude control it can start tumbling due to atmospheric drag, causing flaring behavior of the brightness. There are, however, still exceptions to this analogy as some satellites (e.g. Globalstar's) still show flaring behavior whilst in-use. The benefit of the brightness behavior of satellites is that they can be used as a fingerprint for the object. This can then be used for matching independent observations and also classification.

Classification, however, is a subject that is beyond the scope of this thesis. The code produced by us does contain the option to extract the brightness information for future use.

5.6. Discussion

From the analysis it can be concluded that detectable features are curves rather than straight lines. The most deciding factor in the curvature is the observation angle of the camera, rather than the optics themselves or the satellite's orbit. Due to the size of the PSF of the instrument, bright point sources are often more than a few pixels wide, which causes the tracks to have a width of more than a single pixel. For a line detection method this is beneficial as the curvature is offset against the width of the feature. Therefore the majority of the satellite features have a substantial straight segment within them that is easily detectable by a line detection algorithm.

The most deciding factor in the length and behavior of the feature is dependent on the observation angle followed by the orbit of the satellite (the orbital period and eccentricity). The image frame is large enough to capture complete tracks of lower passing satellites, whilst also being able to observe higher altitude satellites with a detectable feature length. However, for both situations the satellites should be bright enough to be visible in the first place.

This feature analysis shows that for the MASCARA instrument we can determine parameters (valid track length and mean absolute error of the track) to optimize the features we are looking for. These routines can be applied to data from a different arbitrary camera installation, and can output similar findings.

6

Track Detection

This chapter discusses the options for a feature detection method and the motivation behind the final option selected. For the chosen detection method it is explained how the available parameters are fine-tuned based on the results of the feature analysis. Next, to estimate the detection performance in terms of object range, we check against the available information from the TLE catalog. Finally, for a few groups of satellites with accurate TLEs, the detected tracks are sampled and further used for endpoint determination.

Note that this chapter only covers the detection method and the satellite identification and not the unknown object classification. There still exists significant amount of work to be done and options to explore in terms of classification and identification for recurring objects. Remarks are made for results that can aid the classification and identification of objects, similarly to what was mentioned in chapter 5.

6.1. Detection Methods

For detecting features in images there are several options. The application of deep learning for feature detection is becoming increasingly interesting due to the advancement made in field of the machine learning. However when it comes to line detection in particular, there are a few limitations. For instance the input images would have to be subsampled and also a training database would have to be available for this application. Since there was no dataset available of line segments with reference positions the application of machine learning for detection was not investigated. However, the results from this thesis it can be used as a database to train machine learning algorithms in the future.

There are two other methods available for line detection; the RT [22] and the HT [11]. To select the optimal method for feature detection the available options were evaluated. The detection methods that were tried are the following:

- The RT , described in [22]). In its discrete form the RT rotates an 8-bit image by an angle θ and performs column-wise summations of the pixel values onto the projection plane, where the distance on the projection plane is defined as ρ. For each combination of ρ and θ we have a signal response as to how bright that column is, from which we can determine if a line segment is present or not. The implementation of the RT that was used was provided by sci-kit image [32].

- The HT, described in [11]). The HT takes as an input a binary image, and for each positive pixel it performs a set amount of summations. The distance between the positive pixel and the origin is defined as ρ, and the summations are performed for a given number of θ. The summations are done across the line segment that has an angle θ and crosses the whole image. This also provides a response in ρ and θ, similarly to the RT. The version implemented in this research was provided by OpenCV [2].

- And lastly an adaptation of the HT in the Probabilistic HT (PHT), described in [17]. The PHT still runs through all the positive pixels, but now also removes pixels that are part of a detection. Besides the threshold that needs to be passed, the PHT also has two other requirements; the minimum line length (MLL) and the gap length. If these three requirements are met, the line

is considered valid and all pixels taking part in this detection are removed. The efficiency of the method arrives from the removal of the pixels as these do not have to be re-checked. The implementation of the PHT is also available in the OpenCV library.

From literature these discrete methods (and adaptations) are also generally applied in [10, 18, 21, 23, 25]. The RT is oftentimes mentioned to be applied for faint feature extraction and the HT for faster and robust processing. More discussion on this can be found in the literature study that was done for this research [36]. As previously mentioned, the responses from the methods are similar- a more accurate description where these methods differ is described in [33].

6.1.1. Comparison and selection

The methods were compared on a single stacked image to see compare their responses, but ultimately to understand if their implementation worked efficiently and if the outputs were easy to implement within the pipeline. The parameters were discretised to equal values to keep the comparison fair($\rho = 1, \theta = 1$ degree).

Radon Transform

The implementation of the RT left several things to be done as the function is used for backprojection (image restoration) and not line detection. It meant that there is no method available in Python which supports the extraction of linesegments. Furthermore, the implementation was not optimized well and used around 60 seconds per binary frame, whereas the 8-bit image version took approximately 4 minutes per image. The pros and cons are:

+ Able to be used on 8-bit images, allowing potential better detection of faint features.
- No implementation available where line segment positions are returned, the response is only applied for image backprojection and not line detection. This also means that the output for the line segments is only given in angle and distance to origin, meaning that the position on the frame is unknown.
- CPU time is high compared to the HT (60 seconds or longer)

Hough Transform

The standard implementation of the HT returns only the distance and angle of the detected line, thus not the position on the frame itself. This means that it does locate the angle and distance with respect to the origin, however not the exact start and endpoints of the line segment. Also this method required a high threshold to avoid false and double detections. Often wide features would get detected by multiple angles which led to incomplete detections. The pros and cons are:

- Detection method only works for binary images
- Implementation only returns the angle and distance of the line with respect to the origin, it does give line segments but not the exact image location. This would then need another step to extract the position.
+ Low CPU time compared to RT (sub second execution time)

Probabilistic Hough Transform

The PHT implementation has a few parameters that allow fine tuning based on the MLL, threshold and gap length between pixels within the line. This allows tweaking to the features we are expecting. Moreover, the MLL can be chosen based on the findings of the feature analysis. The pros and cons are:

- Detection method only works for binary images
+ Implementation returns the image coordinates of the line segments, which means there is no need for further position determination.
+ Low CPU time compared to RT (sub second execution time)

Based on these findings the PHT was deemed to be the preferred method; it has a low execution time within one second, the feature position on the frame is returned and there is a parameter for the minimum feature length. A low execution time is needed in order to keep the algorithm runtime as small as possible. Since the position on the frame is returned directly, there is no need to further extract the position on the frame. Not only does this decrease the number of operations required, but it also avoids another possible source of error from the line extraction. The extra parameters of the PHT can also be used to optimize for the expected features The MLL can be used for avoiding false detections, however, the maximum gap size for a satellite feature is difficult to define for tumbling objects.

Note that higher order methods were not investigated due to the arbitrary shape that the features might have. If besides angle and distance also shape dimensions should be iterated through (curvature for instance), the CPU time would increase with an expected negligible gain in detection performance as there is always a straight section available. Also the gap length parameter is useful since coincidental alignments of bright sources are not detected as line segments, whilst that could be the case for the other methods. Next, we continue with the PHT for the detection of the features, where we have to set the parameters of the method based on the results from the feature analysis.

6.2. Detection method tuning

For the PHT there are a few input parameters than can be tweaked to improve the performance, these are as follows:

- ρ: the resolution in distance
- θ: the angular resolution
- Detection Length (DL): the number of positive pixels that is needed to flag a feature as a line
- Minimum Line Length (MLL): as the parameter suggests it is the MLL needed of the feature
- Maximum Line Gap (MLG): the limit of the largest gap possible gap of pixels within a detected line feature

The parameters ρ and θ are parameters that can be selected based on CPU time or required detection accuracy. For this application the settings were set at 1 pixel and 0.125 degrees respectively. The single pixel value for ρ was selected as it guarantees that the response is not limited in search width. This means every line segment is one pixel wide. The angle discretisation was chosen such that there was enough overlap for any feature to be detected. We know from the feature analysis that the cardinal directions had a drop-off at about $1,150$ pixels for a width of 5 pixels wide. The angle that this feature makes is about 0.250 degrees, i.e. if we choose a search angle of 0.125 degrees we are sure to detect all of the possible features. Also the discretisation was required to be fractions in the order powers of 2 (half, quarter, eight, etc..) as other discretisation steps introduced some numerical errors leading to false detections.

Since the features we seek in the images can vary in length as well as brightness, we apply the PHT multiple times to search for different types of features with different parameters. The motivation behind this is that we can both look to detect slow and consistent features and longer and fainter features. Slower moving features have a more regular observed brightness since their incoming flux is binned into less pixels, whereas faster moving objects spread their flux out across multiple pixels which might show more variation in signal. After all the sets of parameters have been used we can combine the detections for a more robust feature detection. Based on the results of the feature analysis we define values for the remaining parameters of the PHT.

As an example, the set of PHT parameters for the La Silla South (LSS) camera are formatted in table 6.1.

The lower limit for the DL and MLL we can define based on the rotational velocity of the Earth and the apparent movement of the stars on the frame. Since we correct the alignment of image with the star reference, the remaining hot pixel streaks will be of the length of the star movement. Star movement is greatest near the ecliptic and smallest when looking at the rotational axis (e.g. Polaris). For one stack the time passed is 320 seconds, which is approximately $\frac{1}{270}$th of one day. For a star near the ecliptic this would mean a movement of about 1.33 degrees, or with the pixel scale conversion (table 2.5) about 66.67 pixels in length. Since we don't want to falsely detect the remaining hot pixel

Table 6.1: The set of PHT parameters that are applied for the La Silla South camera. The parameter abbreviations are; Detection Length (DL), Minimum Line Length (MLL), and Maximum Line Gap (MLG). Where the lower MLL is defined by the maximum expected star drift in the stack, and the upper MLL is defined by the straightness limit of features of the respective camera. The detection ratio is a percentage that relates the MLL to the DL.

DL [pixels]	Detection ratio [%]	MLL [pixels]	MLG [pixels]
63	90%	70	16
117	85%	137	29
164	80%	205	41
204	75%	273	51
238	70%	340	59
265	65%	408	66
285	60%	476	71
299	55%	543	74
305	50%	611	76
305	45%	679	76
298	40%	746	74
285	35%	814	71
264	30%	882	66
237	25%	950	59

streaks, the minimum line length should exceed the expected hot pixels streak length. We therefore enforce a MLL of 70 pixels and a detection ratio of 90%, the ratio results in a DL of 63 pixels. Note that this approximation is generic and does not take into account the position on the Earth nor the looking direction of the camera. Both these effects can be implemented in the future if the DL should be decreased and false detections should be removed.

The upper limits for the DL and MLL are defined by the result from the feature analysis and thus is different for every camera. That way we can ensure that for other camera orientations the detection method also works accordingly. The upper MLL is set at the VTL drop-off length, and the DL is defined by a detection ratio of 25%. This was defined such that it would be able to still detect faint and or discontinuous tracks.

For all the intermediate steps between the upper and lower limits, we take a linearly spaced distribution between the lower and upper MLL. The number of samples is chosen such that the percentage between MLL and the detection ratio is spaced by increments of 5%. The MLG was chosen to be 25% of the DL for all sets. This way it also scales with the possible track length of the feature. However there still needs to be a more justifiable value for the MLG parameter.

6.3. Detection Matching and Track Isolation

After the detection step, all the detections need to be matched together as the tracks are often detected multiple times by the same, or by different sets of parameters. This can either be because the track is curved and has two detections at different locations, or that because of the width the track is detected twice. To see how many tracks are in one image we analyze the angle and distance of each detection. Every line segment on the frame have a perpendicular vector; this vector has an angle from the x-axis and a distance from the origin to the line. It means that some lines need to be extrapolated as their perpendicular vector is outside of the frame. This is the same definition as that of the HT and is shown in fig. 6.1.

After the angle and distance of each detection have been determined they are grouped together. First localized clusters are grouped together with a margin that is close to the discretisation level, after which a search method matches potential candidates together. For a standard detection image the response after the localized clustering looks as follows:

The threshold for fine clustering are the mean absolute margin (from table 5.2, e.g. 15.0 for the center camera) for the distance, and for the angle we use 1 degree. After the small clusters are defined, we check the clusters again if they are from the same track. As can be seen on fig. 6.2, for certain tracks there are still multiple clusters present (as there are multiple colors for the same track). To do this we pick a random cluster, and check if there are other clusters nearby that are possibly in-line with

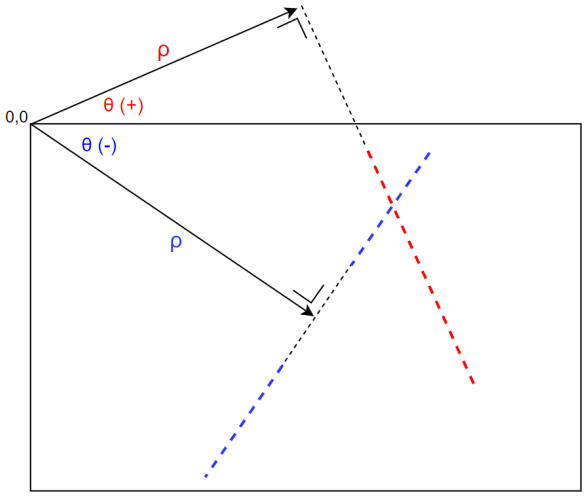


Figure 6.1: The decomposition of two theoretical tracks in one frame. The two blue detections are separated by a gap, whilst the red detection has a perpendicular vector outside the frame. With this definition we can group together detections that are in-line.

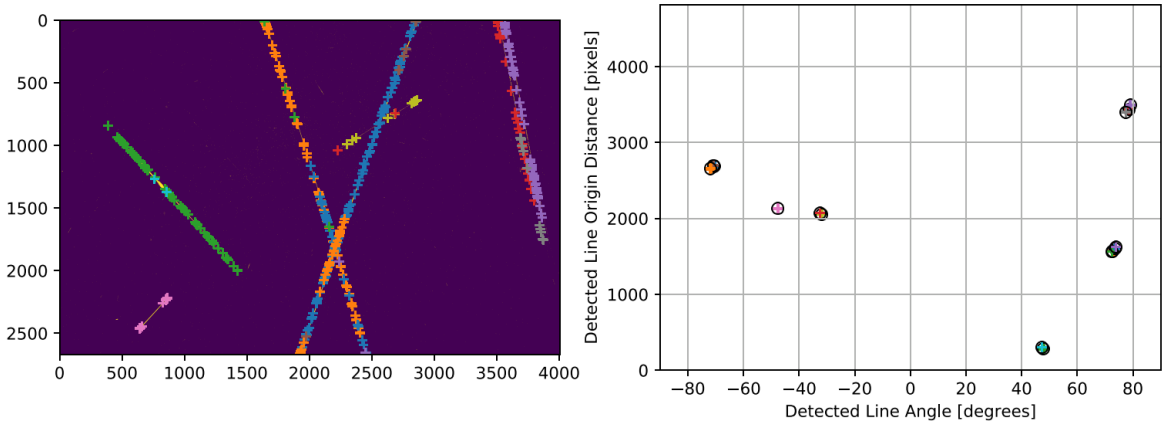


Figure 6.2: The detection image (La Silla South sequence 84) with the detection points from the PHT. The colors of the crosses indicate the cluster that it's in, the colors match between the two images in the figure. It can thus occur that groups of different colors are present in the same track, as the local cluster thresholds are defined close to the discretisation parameters.

the main cluster. This happens by taking the intersect and slope of the main cluster, add the margin we derived from feature analysis, and see if other clusters are within the margins. If there are clusters within those margins, they are thrown in the group and the cycle repeats (new intersect and slope). If there are no candidates left, or the candidates that are left do not fit within the margins, the current group of clusters are labeled as a track and the next cluster is selected. This grouping cycle process is described further below.

1. Grouping of the detections based on the angle and distance (ρ, θ)
   (a) Group very nearby clusters together as they're detections of the same feature, dependent on the following conditions (both)
      - The absolute distance (ρ) to other detections is less than 50% of the track margin (from the feature analysis)
      - The absolute angle (θ) with respect to other clusters is less than half a degree (fourfold of the search discretisation)
   (b) Isolated groups are accepted as a track since they are not a risk for double detections, for this the following conditions both need to be true
      - The absolute distance (ρ) has no other cluster within $10\times$ the track margin (from feature analysis)
      - The absolute angle (θ) has no other cluster within 20 degrees.
2. Iterative searching of the remaining groups
   (a) Select the primary group (first available, random pick)
   (b) Select candidate groups based on nearby angle and distance (averages). Candidates are selected with the following conditions:
      - The absolute distance (ρ) is within $5\times$ of the track margin (from feature analysis)
      - The absolute angle (θ) is within 10 degrees.
   (c) Create a rectangle in the detection image based on the slope and intercept of the current group, with margins derived from the feature analysis table (table 5.2)
   (d) IF a candidate group falls into the rectangle:
      i. Add the candidate group to the current group
      ii. Revert back to step 2.a
   (e) ELSE (no candidates that matches with current group)
      i. Accept the current group as a track
      ii. Remove the current group from the available groups
   (f) Repeat for all groups in the candidates and all groups have been identified as tracks

To conclude, this grouping algorithm allows double detections to be matched together to form one track. Not only is this method able to deal with split detections due to brightness variations, but also with tracks which are curved and thus have different distance and angles.

Note that this method may be flawed for situations where two objects are too close to one another that they are grouped together (similarly to the top right detections in fig. 6.2). However, this issue is common practice in orbit determination and identification as shortly after launch, objects are relatively close and time is needed for the objects to separate. Another option for this situation is to separate the detections manually.

For the future this method still requires some fine-tuning of the parameters, where attention should be given to the thresholds for the distances and angles. In addition, the possible application of a K-means clustering method can be investigated, though this approach would be limited as the number of tracks present is not known a priori. An iterative method where the number of tracks is determined 'on the fly' would be of interest for investigation, however, it might require extensive tweaking and possibly the addition of another parameter to ensure robustness. Currently, we only use the positioning data on the frame, however, the application of the index (thuerefore also time) can also be useful for checking if two segments are actually of the same object.

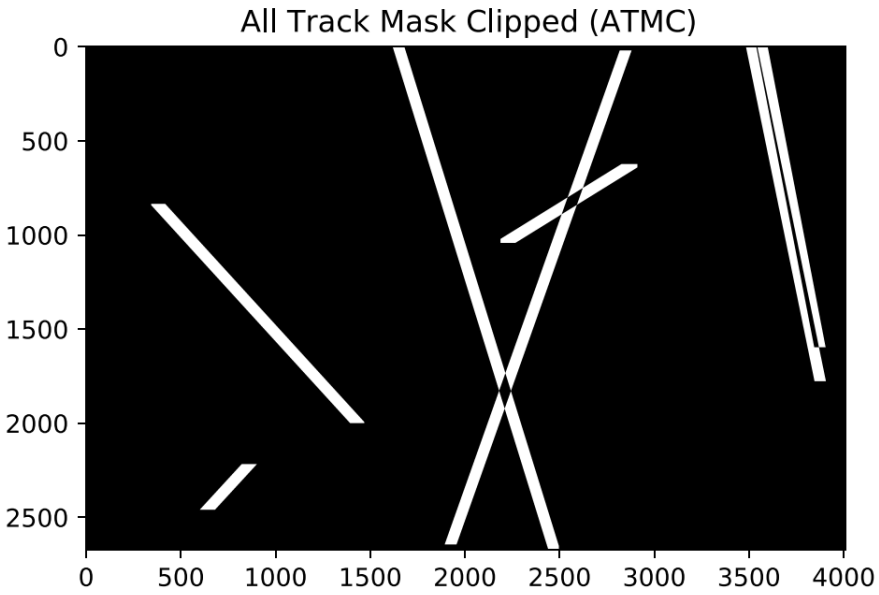


Figure 6.3: The binary All Track Mask Clipped matrix formatted as an image. Note that the crossings in tracks are perfectly removed, and also nearby overlaps are removed.

6.4. Track Isolation

For future processing the tracks are saved with their detection coordinates and the corresponding angle and distance. However the presence of other tracks might cause the track to be obscured by another (if they cross or overlap). For that reason we isolate each track from each other by employing track masks and clipping out the overlaps. To start, each isolated track gets a rectangular mask with the size of the detection range plus the determined feature margin from table 5.2. This is then stored into a new matrix (empty image) where the cells of the masks receive a 1. This is done for all tracks, meaning that the cells in the matrix are cumulative. For all cells with entries higher than 1, the cells are forced to zero as they denote an overlap of one or more tracks. For the sample image shown in fig. 6.2 the All Track Masks Clipped (ATMC) image looks as follows:

To obtain the isolated data for each track, any image can be multiplied with the individual track mask and the ATMC. This is later on used for endpoint determination as the info from other tracks cause outliers in the data to be fitted.

6.5. Satellite Matching

For the satellite matching we apply the individual track mask and the ATMC to isolate the individual tracks. With the same individual track mask, we also isolate all the determined satellite positions, meaning only those within the mask are deemed as potential candidates for the track. For the first detection in the LSS 84 frame this is shown in fig. 6.4.

Next, for each remaining individual satellite ID we create a diluted image of its positions, and then multiply it with the isolated detection image. This gives in return a matrix with where satellite positions overlap with the track. The pixels that remain in the image are called 'bright pixels' since they are satellite positions that correspond to a visible track. The dilution operation on the satellite positions is done for a 5×5 kernel, as that is approximately the track width (due to PSF). The misalignment and the overlapping error due to the difference images is aimed to be resolved by this operation. By dividing the number of bright pixels by the number of individual satellite positions we achieve the brightness ratio per unique satellite ID. For a 5×5 kernel the highest possible ratio is 25.0. The brightness ratio is then one of the indicators for matching the track to a satellite. The other indication is the maximum length of the remaining satellite data points. Thus for a detected track to get a successful match from the catalog these conditions need to be satisfied:

- The length of the available satellite data points is at the least 50% or more compared to the detected track (validation of track length)

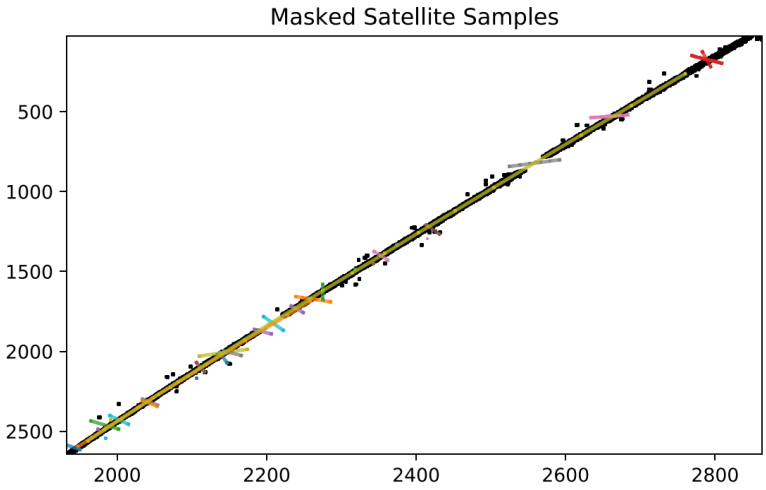


Figure 6.4: For the first detection in the LSS 84 image, the pixels of the restricted track are scatter plotted in black. The potential satellite matches from the catalog are scatter plotted in colors. The axes are not scaled correctly to make the plot size better.

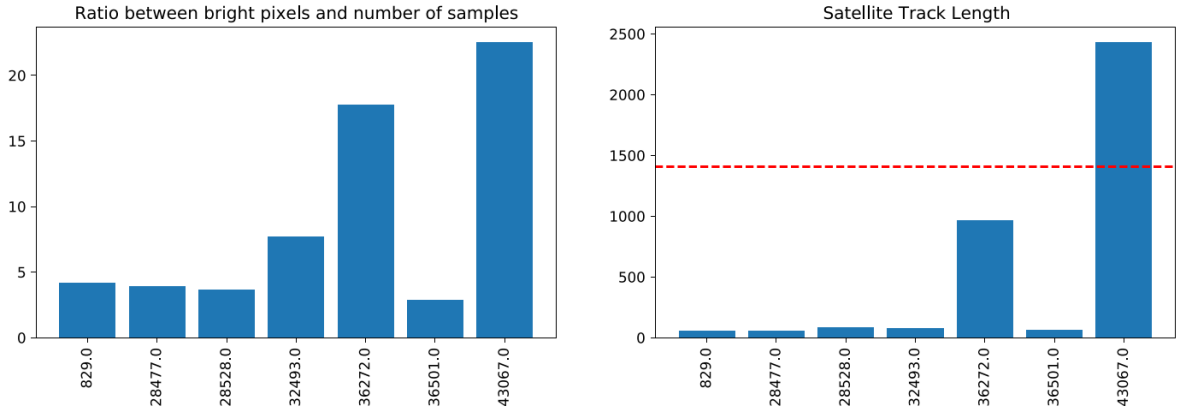


Figure 6.5: Responses for the two indicators for satellite matching. As previously mentioned, the maximum value for the ratio is 25.0, meaning the bright pixel response is adequate. For the track length response, the red dashed line represents the half-length of the detected track, meaning the candidates below this line are already discarded by default. For this case it is quite clear that the track can be matched to the object with satellite ID 43067.

- The ratio between the number of bright pixels and the number of data points is higher than 2.0 (alignment validation). The threshold was selected such that the majority of satellites were removed from comparison.
- If the longest track length and the highest ratio have the same satellite ID, the object is considered a match.

The responses of the track length and bright pixels for the first detection in the LSS sequence 84 stack are shown in fig. 6.5.

This method of matching can be improved upon, especially in situations where the tracks do not completely align with the satellites. Also, for this detection the preference is given to the 43067 satellite, whilst the response of 36272 is also good. By coincidence both satellites overlap with the detected track, however, the first satellite ID has a slightly longer overlap. For this matching result there is thus some uncertainty if the object is correctly matched.

This matching method is more meant for determining if a track is correctly detected and matches well with the information from the TLE. If no match is returned, it is possible that the detected track and the satellite data points are misaligned which causes the bright pixel count to be low. This avoidable by increasing the kernel size, but it means a trade-off with an increased risk of false positives. Additionally, the method can be criticized by its in-track positioning, as there is no check for the positioning along the detected track. The satellite positions themselves don't really matter as long as they are on the detected

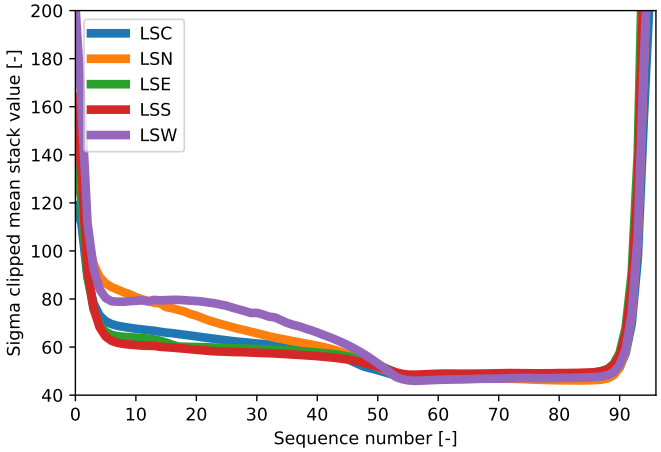


Figure 6.6: The sigma clipped mean of all the created stacks for each of the respective camera directions. The mean value shows an increased brightness at dawn and dusk, but also when a bright moon is above the horizon. The strongest increase is present in the west and north cameras, as the moon was in direct view of those cameras.

feature. For future practice this satellite matching method should be evaluated after the endpoints determination, as it can be matched based on the endpoints similar to FOTOS1. Another option is to check the orbit estimate against objects within this trackmask, where a shortlist of potential candidate orbits can be matched against the estiamte.

The results from this matching algorithm is a database with the detection info of the track and precise positioning info of the satellite that corresponds to the track. This data can be used to train a machine learning in detection as the precision of the method developed by us is quite high.

6.5.1. Flaws of the existing FOTOS1 matching method

When we later compare the performance of the existing method against the newer methods we need to be sure that both detection methods are accurate enough. However, it turned out that the database of the existing method that was used had approximately $10,000$ false matches. This was due to the presence of the bright moon in the west and north camera, as can be seen in the mean value of the background from all the image stacks in fig. 6.6.

The bright moon caused a lot of false detections and by coincidence a lot of false satellite matches. This is because the path of the bright moon crossed a patch of sky where the majority of communications and navigational satellites were present. Since the existing satellite matching algorithm works locally (two endpoints are both within a certain range of two matching satellite positions), the detections were falsely matched to a high altitude satellite. This is by definition not possible for the existing FOTOS1 method as the features of these satellites are not detectable by the method. This was earlier discussed in chapter 5. When we analyze the positions of the feature detections and the positions of the matched satellites from the FOTOS1 method, we can see that the majority of satellite detections occur in the location where the moon passed through the frame. This is shown in fig. 6.7.

A similar result was present in the north camera of the station. To remove these false detections and matches from the results we impose three filters on the FOTOS1 dataset:

- Detections and matches with a perigee altitude beyond $15,000$ kilometers are discarded as they are by default not detectable by FOTOS1 (see fig. 5.4b).
- Detections and matches in the LSW camera are discarded if they have an x-position larger than $3,000$.
- Detections and matches in the LSN camera are discarded if they have an x-position smaller than $1,250$, and a y-position between $[500,1,500]$.

This resulted in the removal of about $10,000$ (mostly false) detections in the FOTOS1 dataset. There are still some potential false detections due to bright features in the evening and in the morning, but

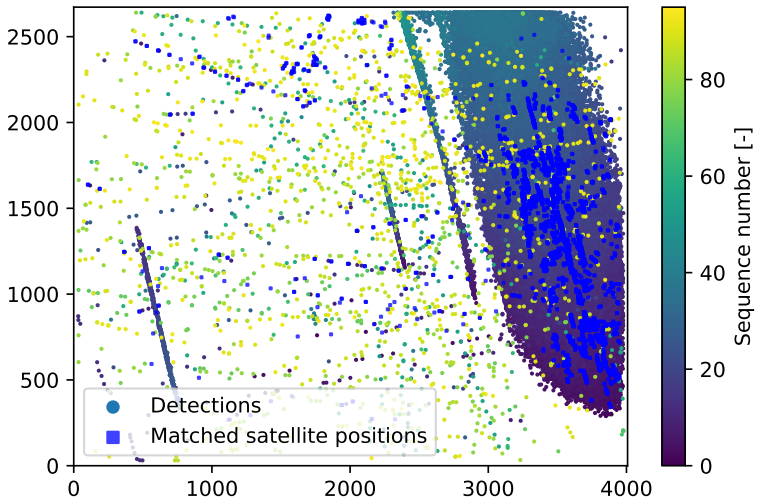


Figure 6.7: The FOTOS1 detections and matched satellite positions in the La Silla West camera. The presence of a bright moon in the evening of the night caused many false detections and also matches.

Table 6.2: The number of uniquely detected objects for different cameras and groups of interest. The data that was used is the La Silla night of 2020-01-03.

Method: group	LSC	LSN	LSE	LSS	LSW	Total
FOTOS1: 0	101	107	173	222	74	421
FOTOS1: 1	47	28	48	64	21	123
FOTOS1: 2	5	5	5	3	9	12
FOTOS1: 3	16	21	21	7	27	49
FOTOS1: all	169	161	254	296	34	605
New: 0	72	121	105	127	25	310
New: 1	19	20	19	27	4	70
New: 2	1	11	11	5	0	26
New: 3	19	50	50	6	5	75
New: all	111	202	158	165	131	481

these are not as significant as those introduced by the bright moon.

6.6. Performance

We compare the detection performance of the new method to the detection results of the aforementioned FOTOS1 algorithm. Since the FOTOS1 method only searched for single streaks, it is not guaranteed that each detected object has enough data points for an accurate orbit determination. The track detection method might be preferred since it is more likely to have more data points from a single detection.

First, we take a look at the number of unique satellites that are detected by the FOTOS1 method and the newly developed method. The matched satellites are determined per camera and per group of interest. The results are formatted in table 6.2.

From table 6.2 it can be seen that the FOTOS1 method detects more unique satellites, however as we know this method is likely to still contain false detections and false satellite matches. The newly developed method performs slightly worse for the lower objects, but is able to detect more unique objects in the higher domain (groups 2 and 3).

To show the detection results more clearly we can plot the detections of both the old and new method together, where the exclusive detections and overlap of detections are color coded. In addition to the scatter plot, also a histogram is made to show the number of unique detected objects over altitude.

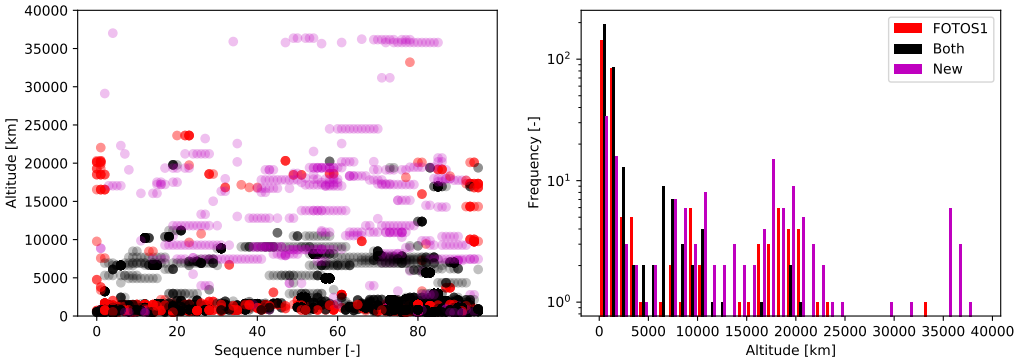


Figure 6.8: A scatter plot on the left that shows the detections of both methods, and a histogram on the right that shows the number of unique objects detected of both methods. Their overlap is shown in black, and the exclusive detections of FOTOS1 and the new method are red and magenta, respectively.

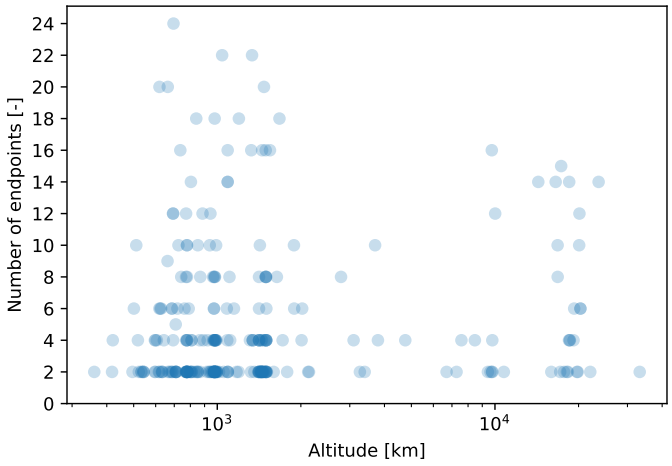


Figure 6.9: For the exclusively detected objects of FOTOS1 we plot the number of endpoints against the object altitude. The transparency of each object is set at 25% to show the cluster near the bottom left of the figure.

These results are shown in fig. 6.8.

What is concluded from the scatter plot is that the new method extends the detectable range far outside the LEO domain. However, the new method does miss out on a significant number of the lower objects compared to the FOTOS1 method. The FOTOS1 method does have sporadic higher altitude detections, but these detections are often rocket bodies with a lower perigee and thus have possible longer features when they are near perigee. The detections that are exclusive to FOTOS1 are expected to be very sporadic detections of short features. We can miss out on these detections with the new method since we look for longer features than 15 pixels in length. When we plot the altitude against the number of unique endpoints for those exclusive FOTOS1 detections we get the result that is shown in fig. 6.9.

From fig. 6.9 we can infer that the majority of exclusive detections from FOTOS1 are singular as they only have two endpoints. Since the duplicate positions are already removed in this data, the presence of mostly even number of endpoints in this figure suggests that the detections spaced apart rather then subsequent. This could possibly be from flaring objects that are only detected at a few instances instead of consistently.

The number of uniquely detected objects is one result, however the number of endpoints per detection is expected to be different. A detection of the new method automatically results into more data points as it detects tracks rather than unique streaks (FOTOS1). The number of unique endpoints per unique objects are also determined and shown in table 6.3.

Table 6.3: Number of unique endpoints per unique objects that are detected. The results are shown per group of interest, where group 0 contains the remainder of the other three defined groups.

Method: group	Mean	SD	Median	Min	Max
FOTOS1: 0	10.06	12.41	4	2	86
FOTOS1: 1	8.86	11.64	4	2	79
FOTOS1: 2	9.00	12.15	5	2	48
FOTOS1: 3	21.47	25.50	12	2	104
New: 0	38.32	76.13	17	3	727
New: 1	26.36	26.12	16	3	147
New: 2	82.62	122.05	31	5	450
New: 3	334.79	467.38	145	4	2480

The number of FOTOS1 endpoints is determined by checking the number of unique timestamps in the FOTOS1 detection data set. We therefore remove endpoints that are duplicates caused by subsequent detections. For the new method we count the number of available satellite points within the detection mask of the track. This is an overestimation as the endpoint determination methods are still required to determine the endpoints from the detection. After the endpoint determination this comparison will be made again, only we compare the determined endpoints instead of the theoretically available endpoints.

What can be concluded from the new detection method is that there is potential for an improvement in terms of number of endpoints. An increase in endpoints can improve orbit determination methods as they function with a lot of uncertainty. However there might be a point where the gain becomes questionable for having more endpoints.

6.7. Concluding

Based on the results, the detection performance is not an improvement in terms of the number of unique satellite detections. The increase in endpoints can however be a good result for the orbit determination methods, however this depends on the performance of the orbit determination itself. If these methods show better estimates by having more data points, the detection performance is an improvement for the objects that are detected.

Based on the information available, it is uncertain what the cause is for the decrease in unique detected satellites. In a general sense the FOTOS1 detections are very local, whilst those of the new method are from a whole track. This means that the FOTOS1 method is more inclined to detect short and bright features, whilst the new method detects consistent longer features. There are, however, still changes that can and will be made to the current pipeline to potentially improve the method.

The satellite matching procedure is quite strict and can be further loosened in terms of accuracy to give more potential candidates. It is also likely that the satellite matching method will be changed to only provide a list of potential candidates, which after orbit determination can be scanned for the (best) matching candidate for potentially updating the TLE. This means that the satellite matching method for the final complete pipeline (including IOD) will have to be re-structured. The output of the current satellite matching method is very accurate as we needed to make sure we get the correct results. The database of detections and the images can be applied for creating a machine learning database, which will likely be investigated after this research.

The detection images also have a some things left to be improved on. There is still the option to remove all the stars to isolate the features better, but this might also remove parts of those features. A possible better way to remove the stars is to process all the difference images within one stack to a median value response, and mask the pixels that are a few standard deviations above the background level. These pixels are likely to be from stars, the moon or other consistent features. We did not explore this option as the stars in the stacks did not seem to be a dominating feature in the images, but will be investigated in the future.

The parameters of the detection method can still be improved upon and could lead to the detection of higher objects. The MLL is now set at 70 pixels, however, this could be refined per camera by using the ASBG. In this image the length of the remnant hot pixel streaks can be estimated, and the threshold can be adjust to that accordingly. This can also be determined from the expected star movement on

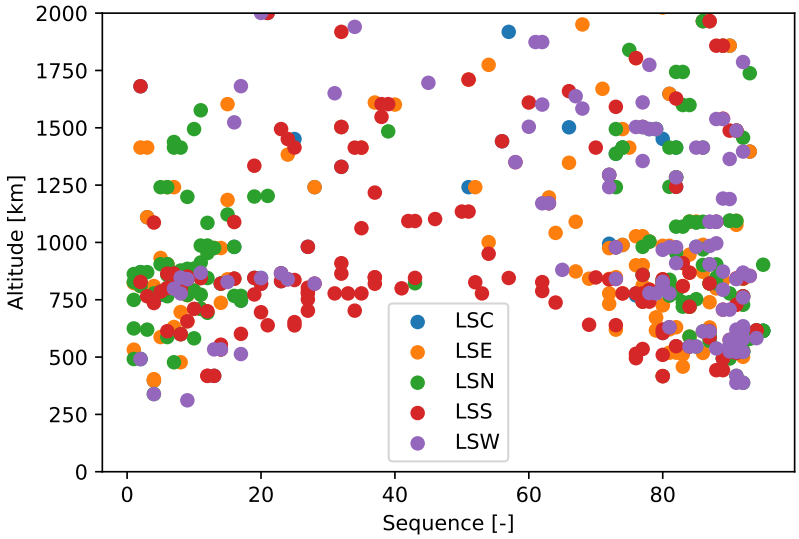


Figure 6.10: The detections for the new method for the restricted altitude range of $2,000$ kilometers. The colors indicate the camera direction.

the frame, as it is what is being eliminated with the AS. The MLG of the tracks is dependent on the brightness consistency of the object. This is not possible to account for as it is an arbitrary effect, thus setting the gapsize parameter is difficult to justify. As of now, the best solution is to set it as a fraction of the threshold, but it should be adjusted to a quantifiable number in the future.

6.7.1. Data reduction correlation

In chapter 3 a comparison was made between the images with expected bright features and the LEO detections by FOTOS1. From the detections of the new method this can also be done, where we only show the detections for objects with an altitude below $2,000$ kilometers, this is shown in fig. 6.10.

In fig. 6.10 we see a similar response to those in chapter 3. Most importantly we again the increase in detections at dusk and dawn. However now we also see the influence a bright moon has on the number of detections in the evening. The southern camera again detects objects during the whole night as it looks beyond the earth shadow. This again confirms the validity of the of the reduction simulation and the potential of using it to reduce the to be processed images.

6.7.2. Number of unmatched tracks

Besides the uniquely detected objects there also have been detections of tracks that were unsuccessful of being matched to an object from the catalog. The failure of being matched can be because of the following reasons:

- It is a false detection of an object that is not a satellite (airplane, lens flare)
- The TLE in the catalog is not up to date and can therefore not be matched
- The match is unsuccessful due to the strict matching method
- The object is not in the catalog (yet)

The number of matched and unmatched tracks per camera are formatted in table 6.4. There are thus still 484 tracks that need to be classified and potentially matched again.

Table 6.4: The total number of matched and unmatched tracks for each camera in the night of testdata. The cause of the failed matching is unknown.

Camera	Center	East	North	South	West
Matched tracks	212	353	482	200	292
Unmatched	26	111	150	95	102

7

Endpoint Determination

In this chapter we discuss the determination of endpoints from the detected tracks. Endpoints are defined as the end of the streaks of satellites, since only those can be used for matching the sky position to a timestamp. Since we have detected tracks the endpoints are now the location where the index changes. Even though we work with tracks the name 'endpoints' will still be used to define the satellite position on the frame.

For IOD we need at the least three unique sky positions with a timestamp such that we can define the state (position and velocity) of an object. Sky positions are given in right ascension and declination with use of the star background, however in this study we stop at just the pixel coordinates on the frame. This is because there are still multiple steps when going from endpoints to orbit estimate, where other errors might also be introduced. Therefore we only evaluate the pixel positions, since that is the highest level of data that is available. The accuracy of the image coordinates directly translates to the quality of the orbit determination, however, this accuracy might plateau due to inaccuracy of the orbit determination method itself. As a consequence, we will only focus on the result from the endpoints compared to the coordinates of the TLE catalog and the SGP4 method.

The existing method for determining the endpoints of the streaklets is done with a cost function that walks along the detected line [35]. This method does not work robustly enough as it is sensitive to objects with varying brightness or noise near the end of streaklets. It also tends to overestimate the positions due to the PSF as features extend beyond the actual position of the satellite. By using a global method rather than a local one, the aim is to reduce the bias of potential local errors without introducing any global errors into the estimates.

The two methods developed are designed such that they determine the endpoints where the index along the track changes. The position where this change happens was estimated with the following two novel methods:

- Double index poly-fit endpoint prediction (DIP)
- Poly-fit Index Trace endpoint determination (PIT)

Both methods will be explained in detail in this chapter, after which the results are compared to the existing detection method. The comparison will consist of:

1. Determining the best performing new method against the SGP4 reference satellite positions.
2. Compare the best performing method against the FOTOS1 method.

It is assumed that the SGP4 points are the true sky-positions, and as described in section 5.1 groups of object types are made to compare different orbit types.

7.1. Double index poly-fit endpoint prediction (DIP)

This method uses the information of the x, y and i data from a detected track to create two second-order polynomial fits in $x(i)$ and $y(i)$, where the i is the discrete time representation in the stack. For

the fitting data we use the index detection image, which is the detection image multiplied by the index stack. This results in an image containing both the track shape but also the movement over time in i. The idea is that we can create two polynomial fits and then predict the position of the endpoints by using the intermediate points. Thus, when we have info for the integer values in time (i), the endpoints occur at the half-integers. Based on the number of indices that are within the fitted data, we define a range of half-integers within that range to predict positions for. The order of the polynomial was again set to be second order, as the relation between index and position did not show higher order effects.

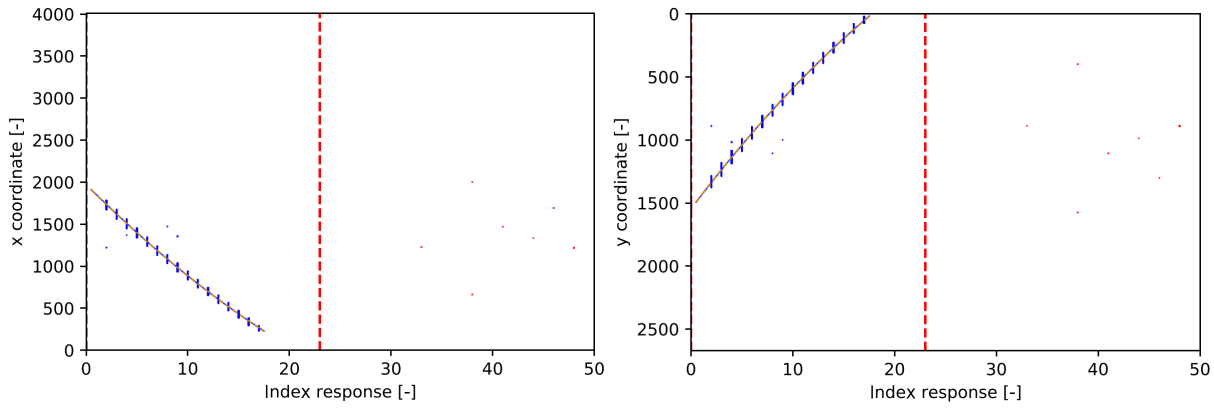
For each respective track, the track detection mask and the ATMC is applied to the detection index image. The remaining data are only positions in x and y and the index i for every pixel.

The errors that are introduced due to the streak misalignment and the gaps between the streaks is still present. However, the method depends on the consistent availability of the data, and since the gaps are also consistently present this is a minor issue. Although it would be an improvement if the gaps were not present to begin with. Another issue present in the data- there are often outliers, either from noise in the index value or from the detection image creation. To deal with these outliers, several regressors were tested to see if they could work robustly with outliers in the data. The comparison was made by the regressors available in the *sci-kit* python package:

- Ordinary least squares (standard, no outlier removal): this method estimates the best possible fit for all the available data such that the squared errors are smallest, hence the name least squares.
- Theil-Sen: regression based on the median values of sub-sampled quadratic fits of the data. By estimating the quadratic terms for a number of data sub samples, the median weights of these responses is less influenced by outliers in the data.
- RANSAC: this method randomly creates a polynomial fit of several datapoints and determines the number of points that agree with this fit (inliers). The fit with the highest number of inliers after an arbitrary number of iterations is returned as the best fit. This regressor was also used in the FOTOS1 study.
- Huber: This regressor fits the data with a loss function that is less sensitive to outliers. For instance the least squares method will always try to include outliers in the fit as they contribute to a high error total. The Huber loss function aims to remove these situations.

From testing it was clear that the Huber regressor performed mediocre as it neglected valuable data and had trouble creating a reasonable fit. Therefore the three remaining methods are kept and also evaluated in the new methods comparison.

An example of the fitting method is shown in fig. 7.1, where both the $x(i)$ and $y(i)$ fits are shown.



(a) The index response against the x coordinate of the fitted track.

(b) The index response against the y coordinate of the fitted track.

Figure 7.1: The data for $x(i)$ and $y(i)$ for a single track. The red dashed lines define the fitting confidence region in i, which the range that is used for fitting the models. The visible lines passing through the data are the fitted polynomials used to predict the endpoints. There are multiple regressors used for the fitted lines, however they are not distinguishable as they overlap.

From the two fits it is clear that the relation between time and position is also not linear. Additionally, because of the oftentimes oversized trackmask, there is some noise within the data. To deal with the noise in the data we apply two confidence regions.

- The first is the confidence region based on the available data that is to be fitted. Only the index values (i) that are within the range of $\pm 3\sigma$ from the mean are considered for fitting the model. This is a very safe filter as it removes only a small portion of outliers.
- The other confidence region is that of what can be predicted. Since the fitting region is 3σ, the prediction region was set to 2σ. This provides the method with sufficient data for endpoint prediction (as one sigma would be too little), whilst staying well within the fitting confidence region.

With this method we accept the risk of throwing away some data ($1 - \sigma$) which could have been used for endpoint determination. The situation can also exist that we predict endpoints that are beyond the detected track, this is however very rare and requires a lot of noise to be present in the data.

7.1.1. Concluding

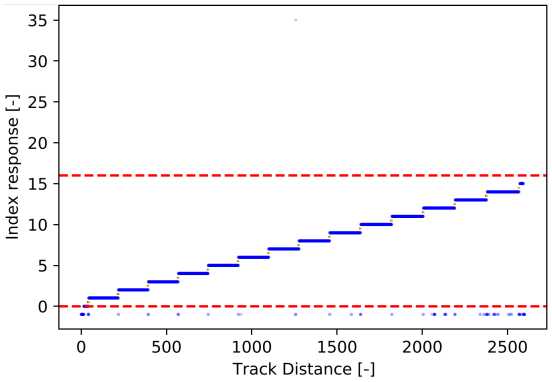
This method is very sensitive to outliers in the data, so we need to ignore parts of the data to make the estimation more robust. Also the risk exists that the double polynomial fit is not a perfect representation of the actual satellite movement, therefore possibly introducing a bias from the polynomial prediction method. This is not only the case globally but also locally. For instance, if there is change in the streak length from image to image, there will be a bias in the fit to the longer streak causing the endpoint to be predicted at the wrong location (closer towards the longer streak). Therefore this method is to be expected to work best for objects that move at a low and more steady speed on the frame. Also the possibility of pulsating features might cause the fit to be biased to the bright flashes (with more data) rather than the actual streaks.

7.2. Poly-fit and index trace endpoint determination (PIT)

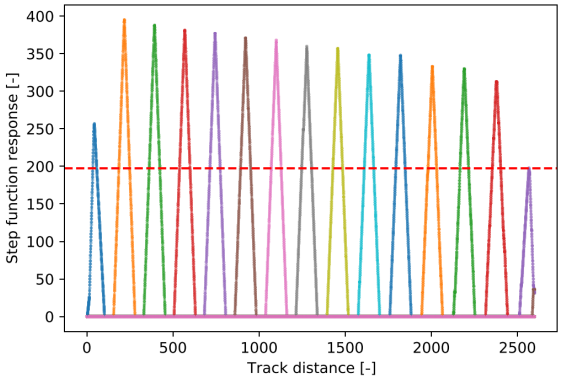
This method fits the track shape ($y(x)$) from the isolated track mask, and traces the given shape in the index determination image to obtain the change in i over $y(x)$. As was shown in chapter 5, a second order polynomial was deemed good enough to fit the track shapes to an accuracy of around the pixel level (well within the PSF). The polynomial fit also removes the misalignment error between the subsequent streaks since the shape is fitted across all the data. Like with the DIP method, the isolated track can still contain outliers in the data. Therefore, we employ the same comparison of regressors, which yielded similar results (Huber performing bad and others having competitive results). After fitting the track shape, the next step is to determine where along the polynomial function, the cross-overs in i occur. Simply said; we need to determine where the index jumps up or down one value (as direction is ambiguous). First we need a definition for distance along the track: s. To do this we discretize the track shape function $y(x)$ into sub-pixel values, after which we can define $s(x, y)$ in Euclidean distance. Then for each $s(x, y)$ we have a corresponding image coordinate, which we can use for extracting the index values. The index values are evaluated if they are increasing or decreasing, this is done by comparing the averages between the first and second half of the index response data.

Next, to determine where the crossovers happen we can not just assume a single increase or decrease of index value. This approach would be highly sensitive to noise and would result in mostly false estimations. Therefore, instead of determining the single value difference between two subsequent positions, it is more robust to convolute a step function over a certain length of the track. The response where the step function convolution is at it's highest can be used as the crossover point. For the highest response the x and y coordinates can be extracted from the $s(x, y)$ data. The gaps that exists between the streaks are also present in the index detection image, therefore this method is slightly more prone to mispositioning the crossover point. An example of the index trace and the responses per crossover is shown in fig. 7.2

The length of the step response s_{streak} is approximated by taking the total track distance of the feature $s(x, y)$ and dividing it by the estimated number of indices in the index trace. The number of indices is determined by summing the number of values per index, and then counting the number of indices that are above 50% of the average. For example, if the mean number of points of an index is 50, indices with more than 25 points are counted. We choose this threshold as there are variations



(a) The index response over the track that is fit by a second order polynomial (least squares regressor) in $y(x)$. The track distance is the arc length along the $y(x)$ curve.



(b) For every index combination a step function was traced along the index response. The convoluted response has a peak where the step function fits the index response best. Every color in this figure corresponds to an index increase, and thus the peaks happen quite consistently. The height of the response also indicate the quality of the endpoint since there is more data available. The red dashed line indicates the threshold when an endpoint determination is accepted.

Figure 7.2: Responses for the PIT method.

present in the number of points per index due to the change in streaklet length. The step function for an increase in index is defined as

$$f_{step}(s, index) = \begin{cases} f_{i=index-1} - 2f_{i\neq index-1} & s = [-s_{streak}, 0] \\ f_{i=index} - 2f_{i\neq index} & s = [0, -s_{streak}] \end{cases} \tag{7.1}$$

In simple terms, all inliers (those that are equal to the target index) as counted as one, and all other values as outliers and attaches a penalty response to it. The penalty response helped in centralizing the peak response to the correct location as noise often didn't affect the response at all. When we convolute the step function for each available index value with the index trace response (fig. 7.2a), we get a convoluted response for each crossover which are shown in fig. 7.2. The endpoint is then returned where the convoluted response was highest. The highest response returns the $s(x, y)$ position for each i.

7.2.1. Concluding

This method is fairly robust as only one fit can introduce biased errors in position. As it uses more info from the whole track it is not prone to single streak fitting errors as in the old method. Since the old method only fitted a single streaklet it could result into diagonal fits across wide features (as the diagonal contains the most inliers for the RANSAC method). The index tracing is also quite robust since the feature width helps in returning the correct index value even if the $y(x)$ fit is not completely correct. Due to the step response convolution this method also does not enforce a bias in terms of time. Therefore, this method might perform better for features that have varying streaklet length and noise in the data. And not only does the convoluted step response indicate the position where the crossover takes place, but it also can give a rating on how good of a quality the crossover is. The height of the peaks in fig. 7.2b give an indication how good the fitted data is. This can possibly be used in the IOD schemes to give a certain confidence to the observations. The effect will be investigated in future work by us as it needs to be implemented within the orbit determination methods, which is beyond the scope of this research.

The downside of this method is that the determined endpoints are integers, rather than floating point predictions like the DIP method. options to improve this method includes determining the crossover point by averaging the pixel position for equal responses in the convoluted response. However, this would also require the discretisation of the step function to be higher as currently the peaks are single positions. It would also possibly resolve the issue of the gaps in between the streaks. Moreover, another improvement that should to be attempted in the future is tracing the original index stack image rather than the index detection image. This might improve the convoluted response by providing more

data to work with, but may introduce more noise. This would also require the subtraction routine to be altered such that the gaps are less present.

7.3. Performance Evaluation

When we estimate the performance of the endpoint determination accuracy we use the SGP4 position as the reference position. As mentioned in section 5.1 there was a desire to create groups of satellites to give a better indication for the performance estimation. These will also be used to indicate the performance for the endpoint determination comparison, as some methods might be able to work better with certain features.

7.3.1. Comparison of the new methods

First we need to determine which of the two novel methods perform best, and which regressor functions more robustly for said method. We approach this by comparing the determinations for which each method has made a valid estimation. Since both methods determine their endpoints in different ways, there are situations where one method might have an endpoint whilst the other method might not. There also is the chance a fitting error occurs from the regressor and there are no endpoints at all. Therefore, the comparison of one endpoint is valid if:

- The DIP method has valid predictions for all three regressors (Least Squares, Theil-Sen, and RANSAC)
- The PIT method also has valid determinations for all three regressors.
- There exists a valid reference satellite endpoint to compare against.

If an endpoint fulfills these requirements, the error for each combination of method and regressor is determined. The error is defined as the absolute Euclidian pixel distance between the estimate and the reference position. No estimation was done to check if there was a systemic error present to the reference positions. It might occur that there are consistent errors among the estimates, perhaps since the reference position is wrong.

As was known from the detection performance, the new track detection method is able to detect a lot more endpoints per detection (see table 6.3), giving this comparison good confidence to select the optimal method to continue with. The results for each of the pre-defined groups are shown in fig. 7.3.

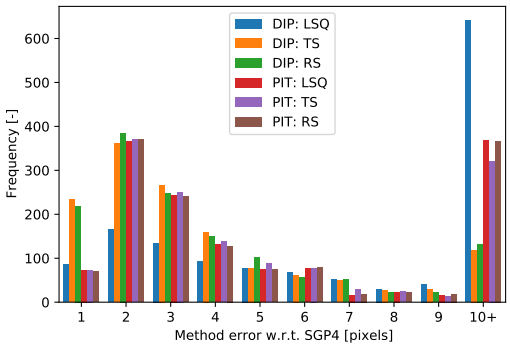
Since the majority of the endpoints lie close to the SGP4 reference positions in groups 1, 2 and 3, it suggests that the large errors are to blame due to bad fitting of the data. This is especially evident for the DIP fit with the least squares regressor due to the high spike. We can conclude from this that the application of the least squares regressor is unwanted for both methods. The preference between the Theil-Sen and Random Sample Consensus (RANSAC) regressor is inconclusive from these results as their errors are virtually the same. However, an issue that was evident from running the scripts is that the RANSAC regressor often threw errors when the fit was not converging to a decent result, whereas the Theil-Sen regressor would return a fit regardless of the perceived quality. Therefore, the preference leans towards the Theil-Sen regressor as it causes less issues during execution.

From all the groups the DIP method performs better than the PIT method. Therefore, we shall use the DIP method with the Theil-Sen regressor for comparing it to the existing FOTOS1 method.

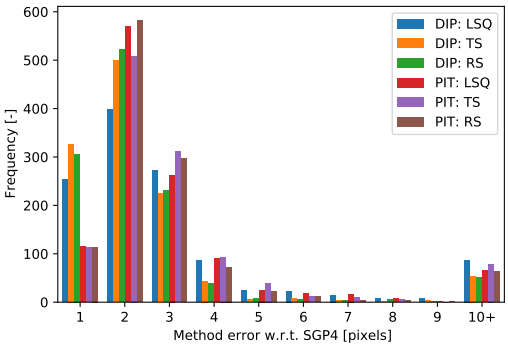
7.3.2. Comparison against the FOTOS1 method

The existing FOTOS1 endpoint determination worked by fitting a RANSAC straight line segment through one streaklet. This line segment is then traced over the original brightness image with a cost-function. The endpoint is set at where the cost-function response is highest. The issue is that the method operates locally and oftentimes fits a faulty straight line segment for wide features. Furthermore, the cost-function can terminate at the wrong location due to varying feature brightness and overestimating due to the PSF.

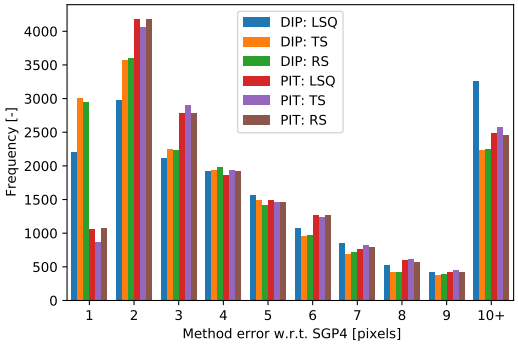
The new method is less susceptible to local feature width since there is more data to create a polynomial fit. Also, the PSF is less of an issue as the feature width is averaged out in the fits. The problem of over estimating the endpoint positions is also reduced for tracks with longer streak segments. This is because the majority of data in the DIP method is from the satellite movement itself and not less from the PSF distribution.



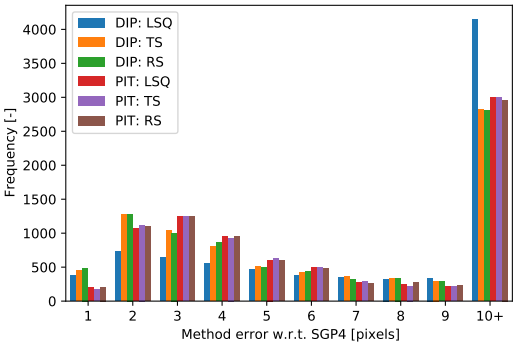
(a) Comparison for the first group; LEO circular (altitude higher than 1,000 km).



(b) Comparison for the second group; circular objects with apogee and perigee above 2,000 km.



(c) Comparison for the third group; rocket bodies with perigee lower than 1,000 km and apogee beyond 2,000 km.



(d) Comparison for all the remaining objects that are detected.

Figure 7.3: The errors with respect to the SGP4 positions for the new methods only. Each error is of a singular point in time and thus a singular endpoint determination. All errors beyond 10 pixels are binned in the same group.

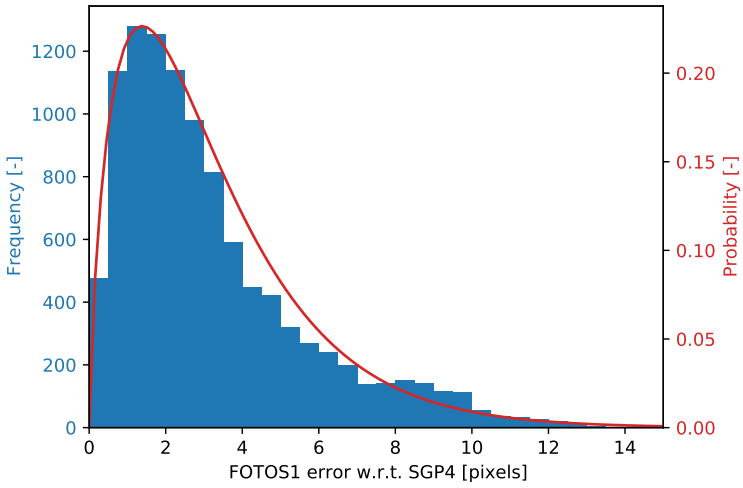


Figure 7.4: The positional error for all of the FOTOS1 cost-function method endpoints with respect to the SGP4 endpoints. The red line shows the Gamma distribution probability density function of the errors.

Table 7.1

	FOTOS1			DIP: TS		
Group	**Mean**	**SD**	**Median**	**Mean**	**SD**	**Median**
0	3.20	2.29	2.83	28.94	133.73	6.32
1	2.56	2.10	2.24	8.31	29.87	3.07
2	1.90	1.63	1.41	6.72	8.08	4.30
3	3.73	2.43	3.16	4.81	7.63	3.61

The error distribution of all the FOTOS1 method determinations is shown in fig. 7.4

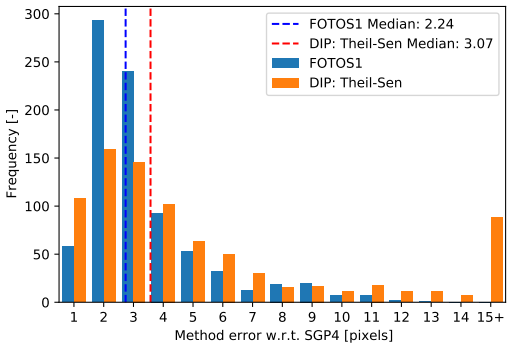
The distribution shows that the method is accurate to about 2-3 pixels on average. However since the satellite matching method works very locally already (within 10 pixels from the SGP4 position), the error can not become too big to begin with. This makes the endpoint determination method a little biased to lower errors.

When we compare the FOTOS1 method directly to the DIP method, the objects in the comparison need to be detected by both methods, thus the objects denoted in black in fig. 6.8. When both methods have valid estimates the errors with respect to the SGP4 position are determined. Note that the FOTOS1 method can have two endpoints per timestamp, as the position can be determined from both directions from two different streaks. For every unique FOTOS1 determination we compare it against the duplicate DIP determination. Thus the errors of the new method can sometimes be counted double. In fig. 7.5 the error distribution per object group is shown.

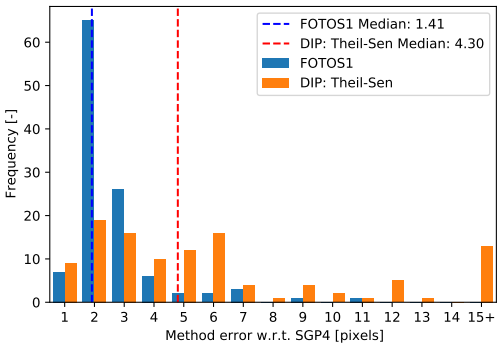
From the distributions and the median lines it seems that the DIP method does not perform better than the existing method, at most it performs to an equal level. Furthermore, since the DIP method is still prone to fitting errors due to noise in the data, the errors beyond 10 pixels are often not useful at all and skew the errors to the right. The error distribution for only the DIP determinations is shown in fig. 7.6. Also, the table with the fitting errors is formatted in table 7.1.

The error distribution looks similar to that of the FOTOS1 method, however, contains a significant number of bad fitting results. This is also clear from the mean and standard deviation of the estimates in table 7.1. What is also evident is that the new method has about double the total amount of data. If we remove the outliers beyond 15 pixels in the Theil-Sen distribution, we can compare the distributions by fitting them with a gamma probability density function (PDF). Within *Scipy* [26], there is a *stats* module where you can regress data to be fit by distributions and thus PDF. Both errors are expressed as their probability instead of their absolute frequencies. This gives a better insight to both methods and how their errors compare.

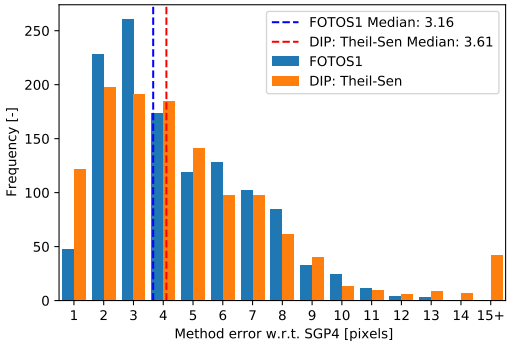
Note that the black line in fig. 7.7 is made from fig. 7.6, but without the errors beyond $15+$ pixels as those would confuse the gamma distribution fitter. From the graphs it is clear that the existing



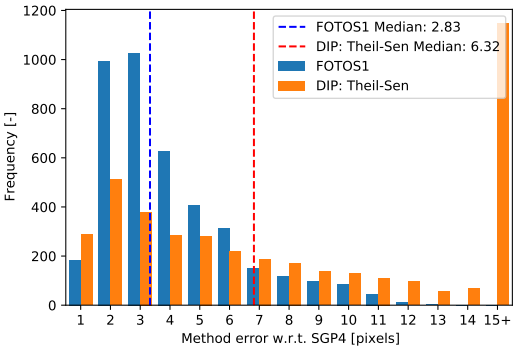
(a) Comparison for the first group; LEO circular (altitude higher than 1,000 km).



(b) Comparison for the second group; circular objects with apogee and perigee above 2,000 km.



(c) Comparison for the third group; rocket bodies with perigee lower than 1,000 km and apogee beyond 2,000 km.



(d) Comparison for all the remaining objects that are detected.

Figure 7.5: The errors of the best performing new method against those of the existing FOTOS1 method. Each error is of a singular point in time of the FOTOS1 method.

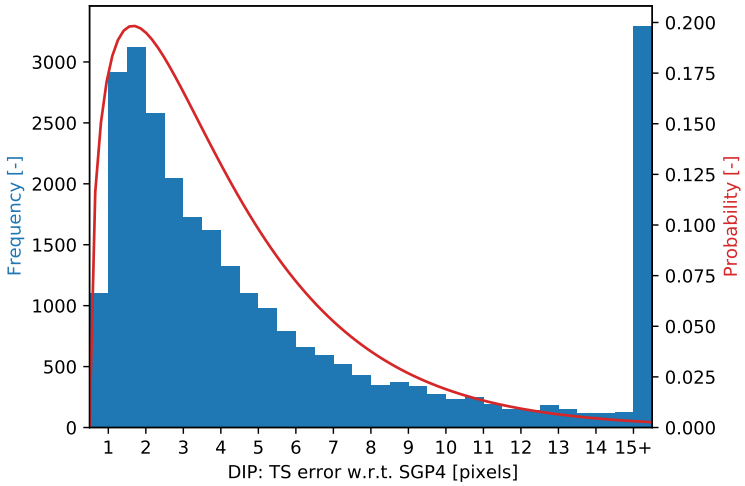


Figure 7.6: The error distribution of only the DIP: Theil-Sen endpoint determinations along with the Gamma distribution fit of the results in red.

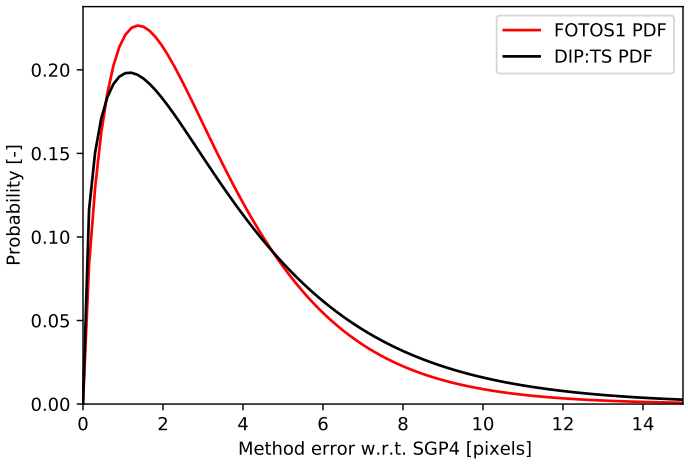


Figure 7.7: Comparing the gamma PDF of both the distributions. The FOTOS1 method has a higher likelihood of getting a sub-2 pixel determination whilst those of the DIP method are more likely to have a larger error.

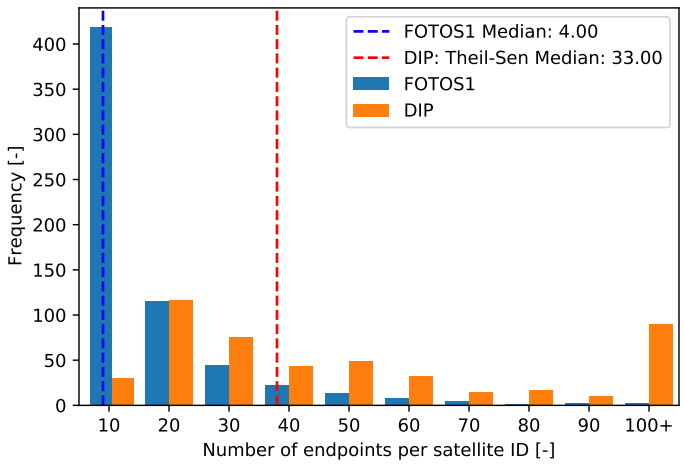


Figure 7.8: The distributions of the number of determined endpoints per unique object.

FOTOS1 method works better (even after removing the far outliers of the DIP method). In relative terms; the FOTOS1 method is better for individual accuracy, but it is important to keep in mind that the new method has close to double the total number of endpoints. The overall distributions of the number of endpoints for FOTOS1 and the DIP: TS methods is shown in fig. 7.8.

7.4. Conclusions and recommendations

The conclusion from the global track fitting methods is that the errors of the methods are comparable to the existing FOTOS1 method. The novel methods, however, show potential and both have improvements that can be made to push the performance some. The recommendations for each methods will be mentioned next.

For the PIT method the greatest improvement that can be made is that the trace should be done on the original index stack image. Also, the subtraction routine change where non-subsequent images are subtracted can help to increase the quality of the index stack image. Especially for slower and thus shorter features there are gaps within the features which introduce false information. Depending on the response the step function might need to be changed to deal with more noisy data. Furthermore, the method could be altered to operate iteratively, where the convoluted response near the ends of

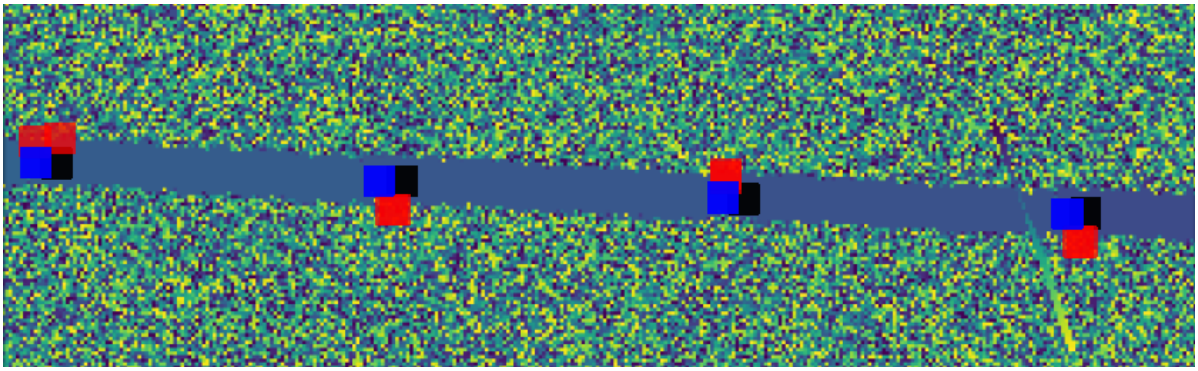


Figure 7.9: An example of determinations of wide features. The FOTOS1 endpoints (red) are compared against the new method endpoints (blue) and the reference SGP4 positions (black). The FOTOS1 endpoints clearly show errors in the cross-track direction possibly due to diagonal fitting of the line segment.

the track can determine if the track mask can be enlarged in the fitted direction. Also, instead of the rectangular track mask, the $y(x)$ fit can be diluted and then used to reduce the noise in the data. These improvements must be investigated in future work as they require changes in all scripts.

The DIP method also benefits from the changes in the subtraction routine, especially for the slower moving objects. Besides that the method can also be altered to operate iteratively by using the two fits to select new data and perform new fits. Next, the fit range can be extended to see if more data is found in the index range.

A general improvement to the analysis of the endpoint determination methods is the error analysis in terms of the cross-track and in-track directions. This would give more insight into possible differences in error determination. It is expected that the errors of FOTOS1 are mostly due to the PSF, where they could be from fitting a diagonal line and/or from terminating at the furthest end of the streak. An example of the diagonal fitting error of FOTOS1 is shown in fig. 7.9.

Further evaluation and improvement of these two novel methods are likely to increase the accuracy. Also, the evaluation of the cross-track and in-track errors can help to get some needed insight in where the methods are introducing their errors. The true performance of these methods has not been extracted yet, but the mentioned improvements are likely to increase that.

8

Conclusions and recommendations

To conclude this research we go back to the research question and goals that were stated in chapter 1. We start by discussing the sub-goals after which we answer the main research question.

8.1. Reduction

This sub-goal was defined to potentially reduce the number of images that are to be processed whilst taking into account the likelihood of potential features in the images. The sub-goal was formulated as:

For each station, date and satellite altitude range, what are the useful images for each night such that only images are considered are most likely to contain sun-lit objects and thus satellite streaklets?

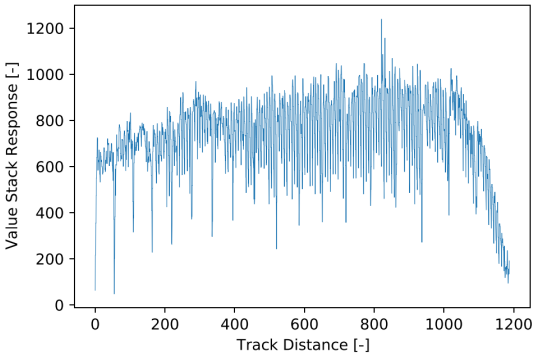
When applied to a LEO restriction the data reduction simulation accurately determined the list of images of interest and also showed strong correlation to the detections of FOTOS1 and the new method. When applied to LEO the potential decrease in images-to-be-processed is significant for longer nights (48%). We can therefore conclude that this sub-goal has been achieved when applied to LEO. However when dealing with higher altitude limits, the reduction simulation becomes less relevant as more objects are sunlit during the night. However the simulation can still be used to give a percentage for each respective image on how likely it is to contain a sunlit object. If it turns out that there is no need to remove any images from processing there are alternative uses for this simulation, this was also discussed in chapter 3. This can for instance be for defining the expected feature length within the images or observation tasking if a single camera on a mount is used.

8.2. Feature detection

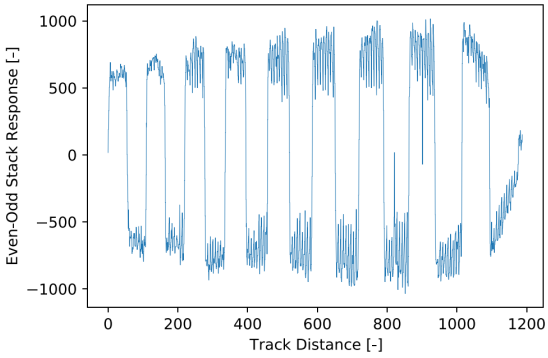
This sub-goal was defined to combine the images for detection and optimize the detection method for the expected features within the images. It was defined as:

What is the optimal method for combining and processing the astronomical images for robustly detecting any satellite feature, taking into account the potential shape of the expected features?

A main objective of this thesis was to combine the images together to potentially detect more objects. Combining the images together highlighted longer satellite features, which was the goal. Comparing the detection performance in terms of unique objects, the FOTOS1 method was found to be more successful. We argue that the level of noise that comes with stacking all the images together can explain our method's performance compared to FOTOS1 which would naturally contain less noise. Also the detection length for FOTOS1 was significantly shorter than that of the new method, 15 pixels compared to 70 pixels. This probably also resulted into FOTOS1 being able to detect a lot more unique objects with mostly 2 or 4 endpoints per object. Comparing in terms of the total number of endpoints indicated that the method produced by us was superior. This makes sense as our stacking method provides long features with many potential endpoints, whilst FOTOS1 operates by detecting single streak objects.



(a) The response along the PIT function, but instead of the index response it is the response from the value stack.



(b) The response along the PIT function, but instead of the index response it is the response from the even-odd stack. It can be seen when the crossovers happen between the images as the value passes through x=0.

Figure 8.1: Trace responses for possible fingerprinting of the detections.

In discussing the differences between the two methods of feature detection, it must be noted that the FOTOS1 method actively removed the stars, unlike our method where the overall detection length was long enough to not falsely detect stars. Although our method showed that removing stars could be avoided, star removal is a promising operation that might be good to investigate further. However instead of removing all the stars based on information from a catalog, we can use a median value stack from all difference images. The median response from the difference images should contain the stars and moon features whilst not containing the satellite features. In hindsight, one can argue that this should already have been implemented in this thesis, so it is highly recommended for future work.

This research aimed to provide an improved method to feature detection, however did not necessarily do this as the number of unique objects decreased. Assessing the methods' performance depends largely on how many data points are needed for the orbit determination methods to work well. From findings in [35], using only three datapoints is insufficient for an accurate IOD. Therefore, the increase in endpoints per detection is welcomed, but it is unsure how great this would improve the orbit estimate. Also, since the FOTOS1 detection method worked better for lower passing objects, the detection approach might be changed such that at the beginning and end of the night the stacks are created from less images, whilst in the middle of the night a stack of 50 can be used. We can use the detection reduction to create an estimate on how many images can be stacked to obtain features of similar length.

The current implementation of the detection algorithm is still incomplete as there is nothing written that classifies what type of object the detection is. We currently rely on the presence of accurate satellite positions to match a detection to a satellite ID. However when there is no satellite that matches the track the track is classified as unknown. To help classify the object that is detected we can use information from the brightness of the track. As an example, instead of tracing the index image we can also trace the value stack and even-odd stack images. For a given track the response could be used to help identify the object, which possibly makes matching multiple overflights possible. This is, however, very challenging as there are still a lot of effects present in the track that are unresolved. As an example the responses for a Ukrainian rocket body (NORAD ID 40879) are shown in fig. 8.1.

For more accurate traces we can also use routines from the current FOTOS1 code to extract the calibrated magnitude of the feature. Also the periodicity of the brightness can be used to filter out false detections of airplanes as these show blinking patterns at a fixed cadence. What is however tricky about the variation is that it is given in terms of track distance and not in time. The best we can do is of course the discrete time representation, however this might not be accurate enough to relate it to a potential rotation model. It should be investigated if a linear movement assumption is good enough for this application. The general application of brightness tracing is recommended for future work as it can help in classifying the detections.

In general the whole detection and classification method needs to be streamlined with the endpoint determination routine as several results can be used interchangeably. For instance, after detection we can trace the result and extract information for endpoint determination but also classification. If it turns

out that it is not a satellite we don't have to continue, whereas otherwise we can possibly iterate.

8.3. Endpoint determination

This sub-goal was defined to improve upon the existing FOTOS1 method for determining the endpoints of the detected streaks. Since we now detect tracks instead of single streaks, we can also use time data for the determination. The goal was defined as:

After the features have been detected, what is the best method for determining the endpoints by using information from multiple discrete moments in time to achieve an overall better fit with with greater accuracy and that also works robustly?

Two shortcomings of the existing FOTOS1 method is the straight line assumption and the cost-function tracing. The way the FOTOS1 method fits a line through the detected streak is biased to fit the most pixels along a straight line. Short and wide features start to become rectangles, and the longest distance of a rectangle is the diagonal. This introduces an inherent bias in the endpoint as they are off the center-line of the feature (this was shown in fig. 7.9). Another issue is that the cost-function does not account for the PSF of the instrument. The cost-function determines that the endpoint is where the feature stops becoming bright, which puts it at the outer end of the feature. This is incorrect depending on the PSF, as that might cause an error of about 2-3 pixels.

The two novel methods developed aimed to avoid this by using all the available data from the detected tracks. The straight line assumption was replaced by one or two polynomial fits which aimed to remove the cross-track error. The PSF was not corrected for, but since endpoints are determined in crossovers of data the PSF is consistent and thus in theory should be canceled out. From the results it was evident that the double index fit prediction method worked better than the index tracing method. Not only were the errors smaller but also it worked more robust as there were less large errors. However, when comparing this method to the existing local determination method, the performance was not an improvement. The oftentimes noisy input data for the fits caused the endpoint determination to be off by a large margin.

Both methods were therefore not very robust and thus they might gain a lot of performance by iteratively fitting the data and removing the outliers. The improvements to be made were already mentioned in chapter 7.

Another take-away is that the methods currently only work with MASCARA's continuous observation strategy, and still needs some effort to be compatible with the gaps in bRing's observation strategy. For the bRing adaptation the effect of the PSF also becomes an issue again since there is no overlap of streaks anymore. Also the index tracing method would have to be changed to something else than a step function as the trace would alternate between noise and correct indices. The double index prediction method could still work normally but would perform less accurately since there is less data to work with.

8.4. Research question

The main research question encapsulates all the aforementioned sub-goals and more. It was formatted as:

To develop an efficient automated satellite track detection algorithm that is able to detect satellite tracks of objects in the LEO domain and possibly higher, by way of stacking images, image processing operations, feature detecting methods and fitting the data with an overall fit, to be able to more robustly and accurately determine the endpoints of streaks and thus indirectly improve the performance of the orbit determination algorithms.

So, not only does this research question contain all the aforementioned sub-goals, but it also combines them to form an efficient processing pipeline.

Efficiency in this research is quantified in terms of the total processing time, and if that processing time allows all the data to be processed within 24 hours. However, since the pipeline is not complete, as classification and orbit determination are missing, we can only give an estimate as to how long the image processing part of the pipeline will take. The scripts that were written and the order of execution is shown in appendix B. Within the overview the description, time estimate and improvements of all

scripts are given. Since all the scripts were run on the server of the University of Leiden, they ran significantly quicker compared to a local executable. The majority of time spent in the scripts is from file handling (importing and exporting of intermediate files).

What is concluded from the estimates is that processing all the data from one camera takes approximately 17 hours. This estimate is based on having no time loss for intermediate file handling, thus all operations are performed within the same script. The majority of the time needed is for the creation of stacks and the satellite catalog calculations. Where the latter can still be improved by possibly filtering out slow objects.

For the pipeline it is thus almost a requirement that the scripts can be run whilst observations are taking place. If the processing can only be done during the day, a lot of data will have to be filtered out due to lack of available processing time. If it is indeed possible to process during observation, the goal of processing all the data within 24 hours can be reached. However it does also depend on the computation time of the orbit determination method.

The pipeline is able to run autonomously on the current night of MASCARA testdata, however, it still needs to be tested on more challenging nights. The user can optionally define a limiting altitude, but this is by default set to having no limit. Also the pipeline can be adapted to operate with potential other observation strategies if the input files are similar (images and AS). There are also some parameters that can be changed dependent on the user's need, for instance how many line searches should be done or how secure the satellite matching should be. There is, however, a need for human intervention after the orbit has been determined to validate if the procedure was done correctly. This can for instance be for detected objects that don't have a satellite match, be it either a false detection or a potential new object.

In terms of accuracy of the satellite track endpoints, the new methods are not an improvement. The accuracy is at most comparable to that of FOTOS1, whilst also being prone to fitting errors. There are some improvements that can be made to remove noise from the data, this would possibly lead more robust fitting of the polynomials.

When looking at the results from this research it can be concluded that a stacking method for detection might not be the ideal solution in terms of detecting unique objects, however, it is a good solution to processing the images efficiently and also obtain more data points per detected object. The endpoint determination methods were not necessarily better as the mean accuracy for all three groups of objects were worse than the existing FOTOS1 method. A benefit of the detection and endpoint determination methods is that the increase of endpoints per object is good for applying batched methods for orbit determination. This might lead to better orbit estimations for the detected objects, but remains to be investigated.

8.5. Recommendations

During the thesis many suggestions were given for future improvements. The most important ones that also require extra explanation are mentioned in this section.

8.5.1. Subtraction routine change

As was said in chapter 4, there is a potential need for changing the way images are subtracted from each other. With the current subtraction routine subsequent images are subtracted from each other. Due to the PSF there is an overlap in the streaks from subsequent images. Consequently, there will be a gap in between the two streaks in the difference image. This can very clearly be seen in fig. 8.2.

The gaps occur since the PSF bleed is subtracted from each other and essentially cancels out the effect of the PSF. The location of the crossover is likely to be within the gap, however, since it now consists of noise there is no index information in the gaps. By subtracting images with one exposure gap the streaks will all still have the PSF bleeding at the ends of their streaks. When these difference images are then stacked there should be no gaps within the tracks and the PSF bleeding will be removed due to the brightness change rather than the subtraction.

8.5.2. Changes in reduction simulation

There are two small changes that can be implemented in the reduction simulation. Besides the possibility of changing the application completely, the phase angle still needs to be taken into account in the simulation. As was mentioned in chapter 3 the sunlit points percentage drops but the phase angle of

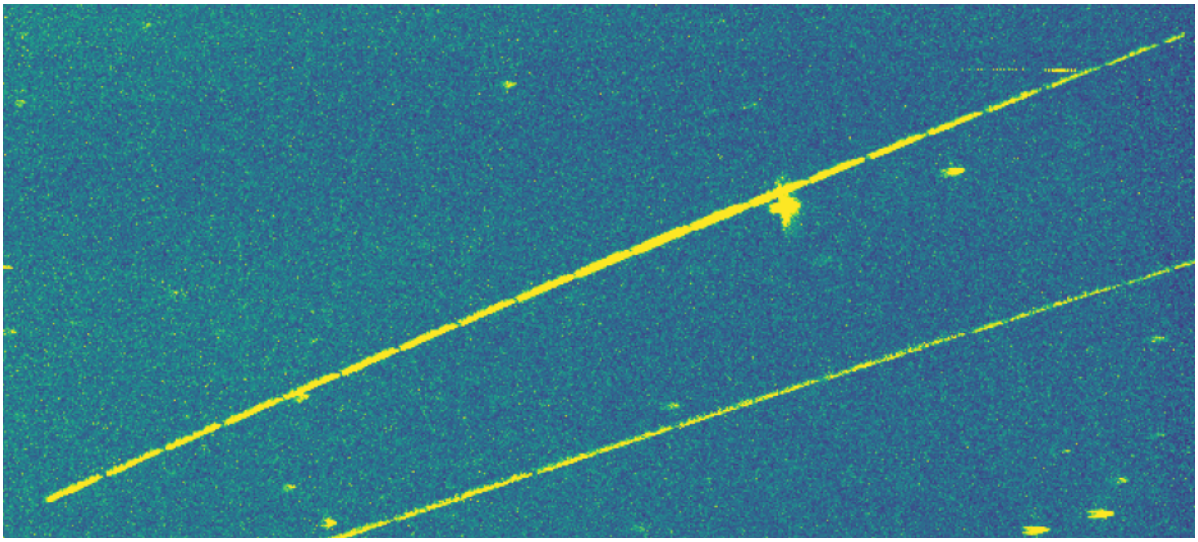


Figure 8.2: Gaps in between streaklets for a whole track. The track is an arbitrary satellite from the Northern La Silla camera.

the points increases. Instead of only using the sunlit points percentage, a multiplication with the phase angle can be implemented. Similarly, also the distance between the observer and the points can be added such that it mimics eq. (5.4) that was mentioned in chapter 5. This would give a more accurate estimate for the possible presence of features within one stack.

8.5.3. Machine learning applications

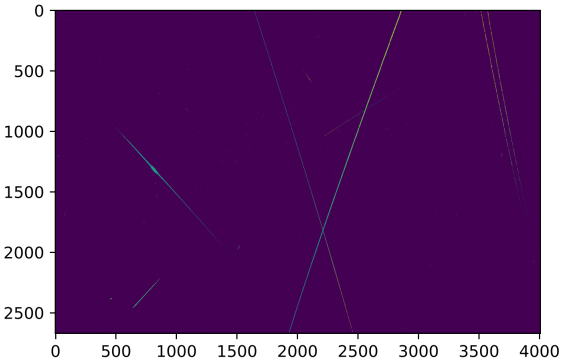
Some scripts or functions in this pipeline can be replaced by a machine learning application. These are the following;

- Feature detection
- Feature detection isolation
- Object classification and satellite matching

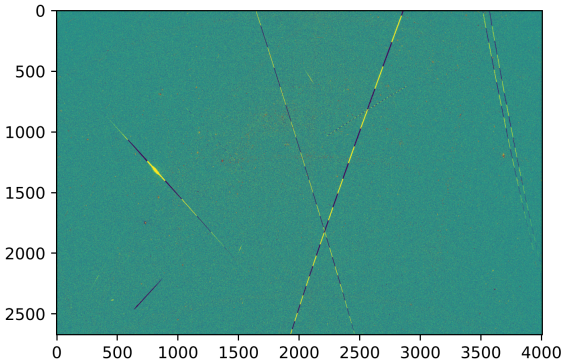
For feature detection it was decided that deep learning methods lack performance when it comes to straight line detection. However, some products that were made might be better for deep learning detection. Deep learning for feature detection usually works very locally, therefore the features that are presented in the Even-Odd value stack (fig. 8.3b) and the detection index stack (fig. 8.3a) might be better input products for feature detection. Within those products we are not looking for long straight features anymore, but rather local features that stand out among the noise.

As mentioned before in chapter 6, the detection clusters need to be grouped into separate tracks like was shown in fig. 6.2. Currently the cyclic grouping method is able to do this, however in machine learning there exists several solutions for grouping clusters together. This should be investigated in the future as it might prove to be more robust compared to the current solution. The difficult thing is that the separation of the parameters are not consistent. In one image the clusters might be very close together and in another image the clusters might be spread out more. Also it is unknown how many unique tracks are present in the image, thus the method has to adjust the number of clusters based on the data itself.

Finally the application of machine learning can be applied for classifying objects from their lightcurve responses, these were previously shown in fig. 8.1. As said before the responses can be used as a fingerprint for object identification or distinguishing between object type (satellite, plane, etc.). The classification of objects with lightcurves is something that was already investigated in [19] with success (96% accuracy between satellites and planes). The addition of more data in the trace responses could make this a very robust classifier.



(a) The detection index stack, it is the product of multiplying the index image with the detection image. Features in this image have a consistent increase or decrease in index value against a background of zeros with some noise.



(b) The Even-Odd value stack, it is the product from subtracting the odd value stack from the even value stack. Satellite features stand out against the background since their pixel values change from positive to negative in subsequent streaks.

Figure 8.3: Potential interesting input products for machine learning feature detection.

A

Image Processing Steps

This appendix is a visual aid to the procedures mentioned in chapter 4. For the overview of image processing steps we make use of two images; a stack from the La Silla South camera, and a stack from the La Silla West camera. The first one is used to illustrate the general image operations, the latter is used to illustrate the moon masking procedure.

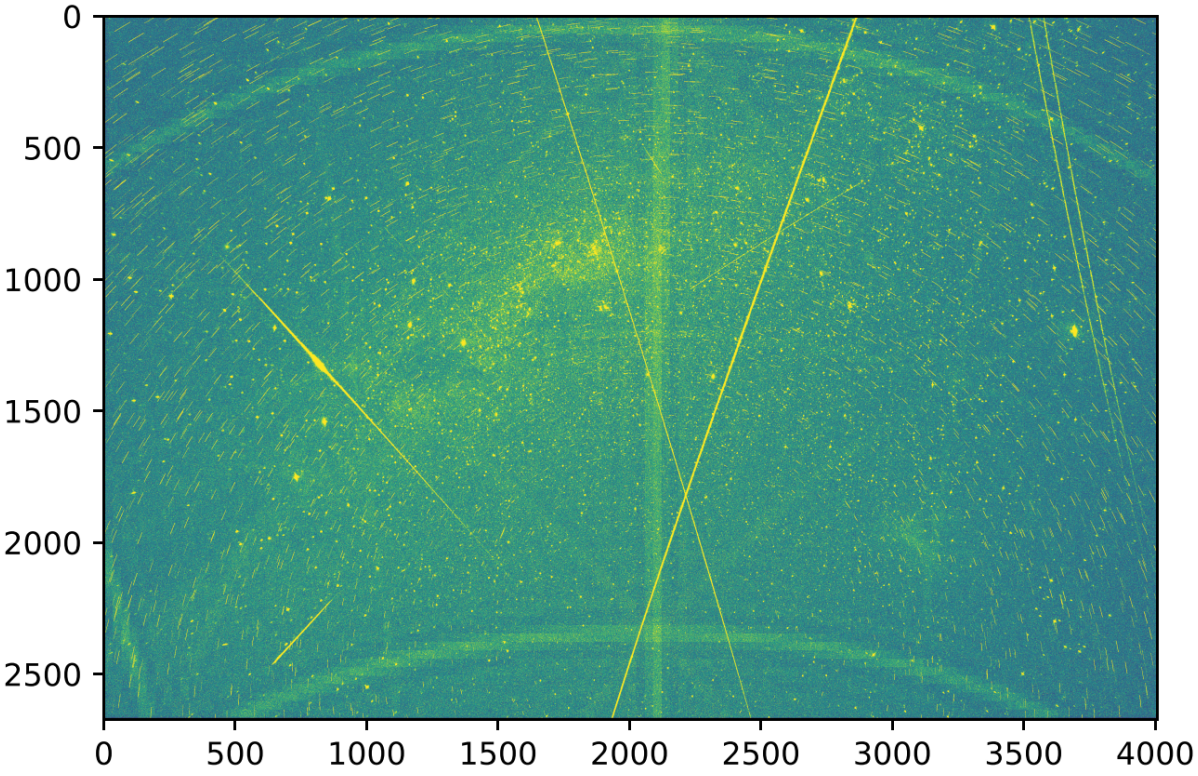


Figure A.1: Uncalibrated stack image for La Silla South (LSS) for 2020-01-03, stack number 84 of the night.

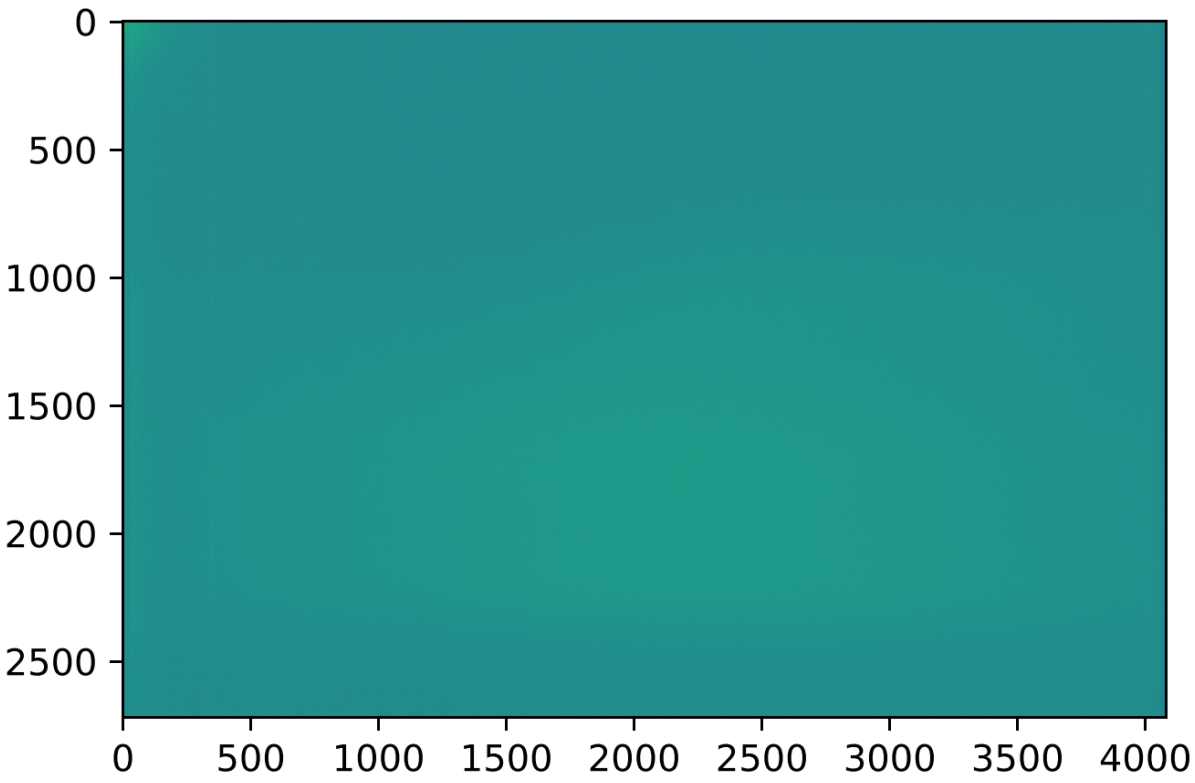


Figure A.2: The master dark calibration frame for the La Silla South camera for 2020-01-03. The image consists of the median values over 40 darkframes.

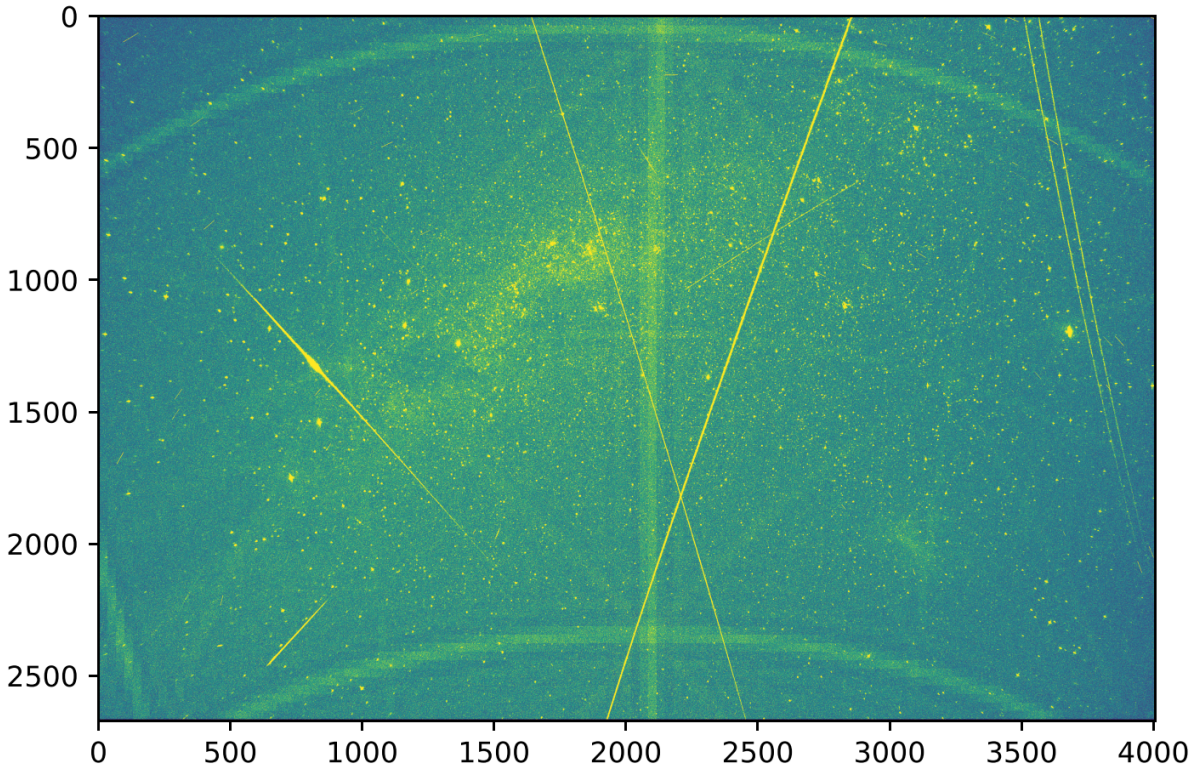


Figure A.3: LSS84 calibrated stack that is a result from subtracting the master dark from each of the science frames.

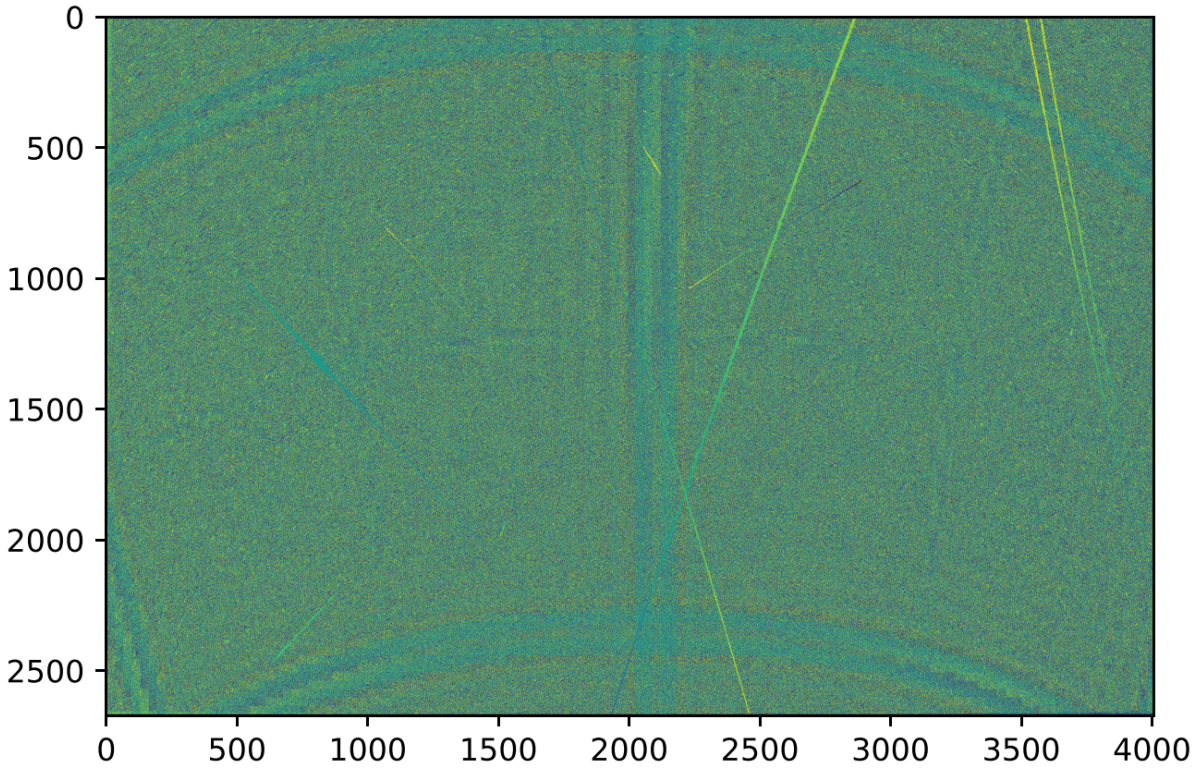


Figure A.4: LSS84 index stack

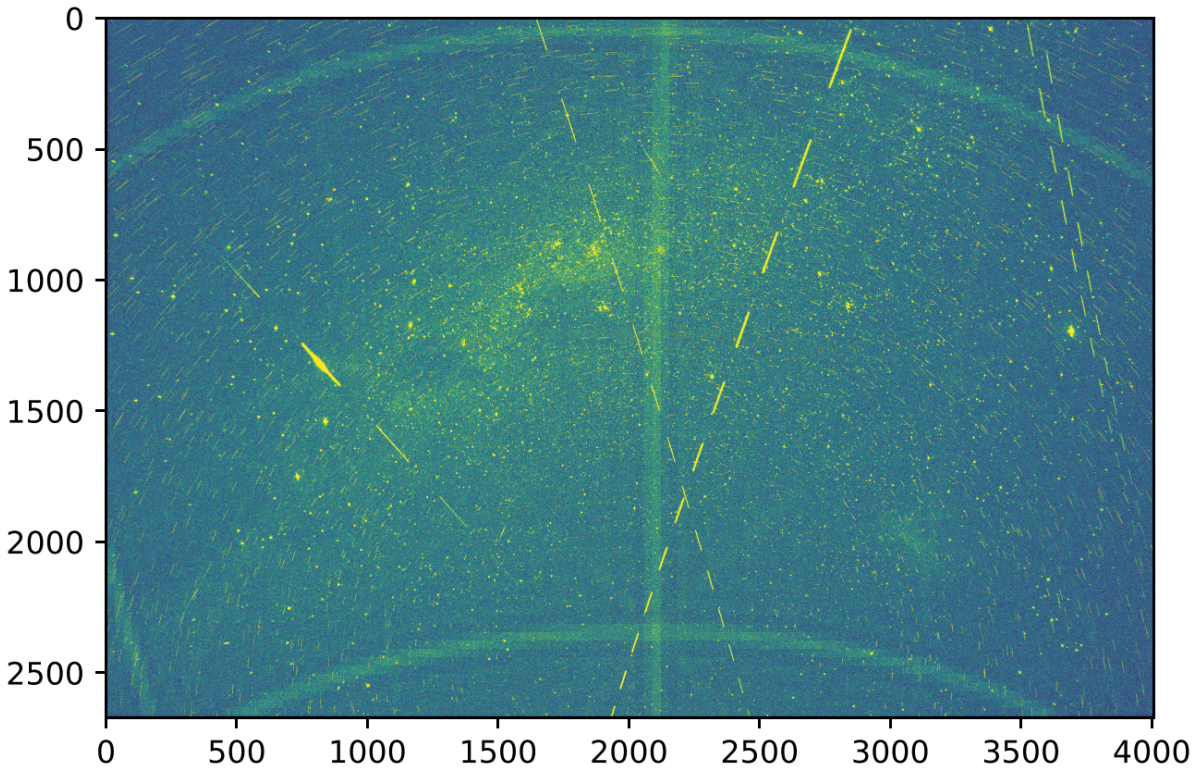


Figure A.5: LSS84 even stack

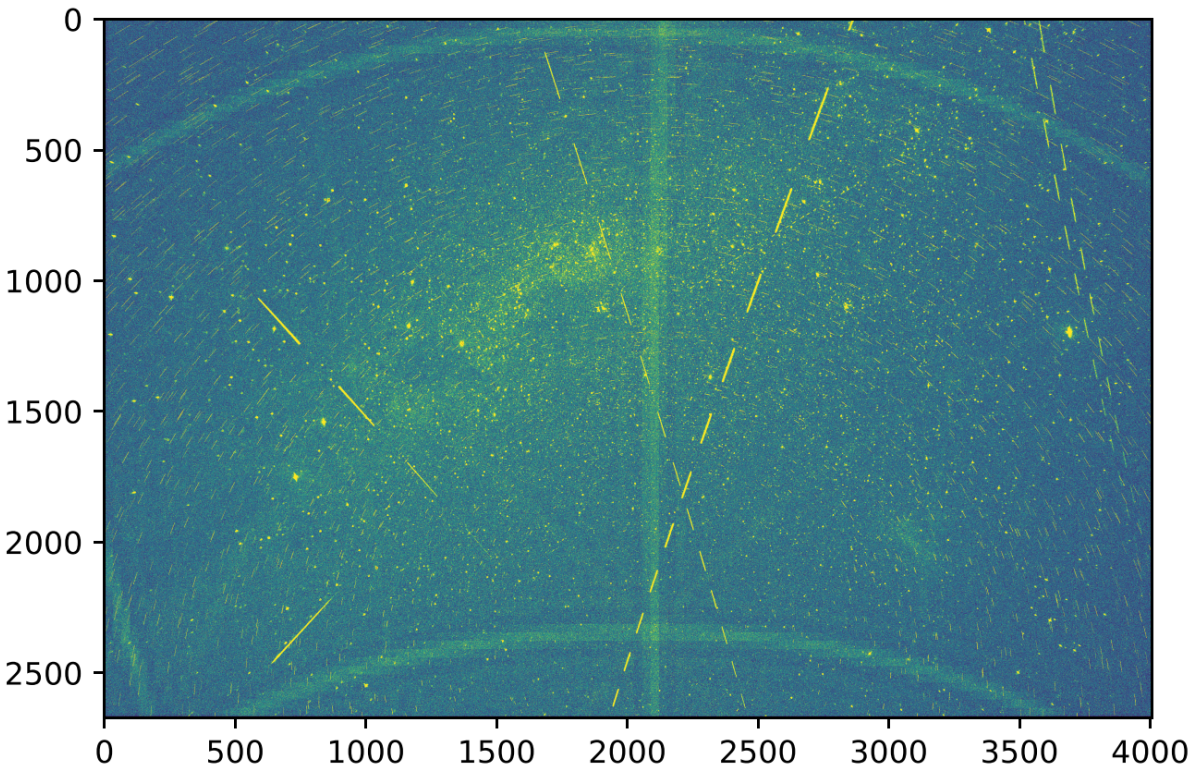


Figure A.6: LSS84 odd stack

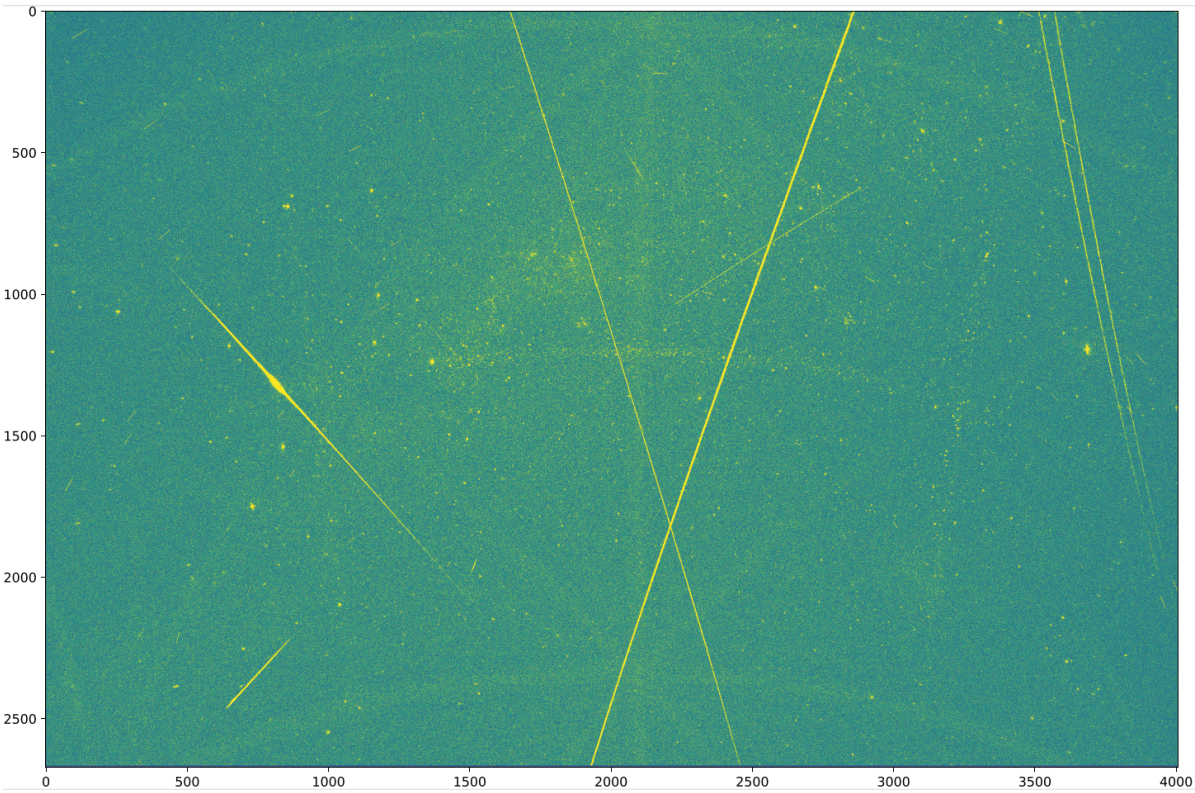


Figure A.7: LSS84 even odd difference

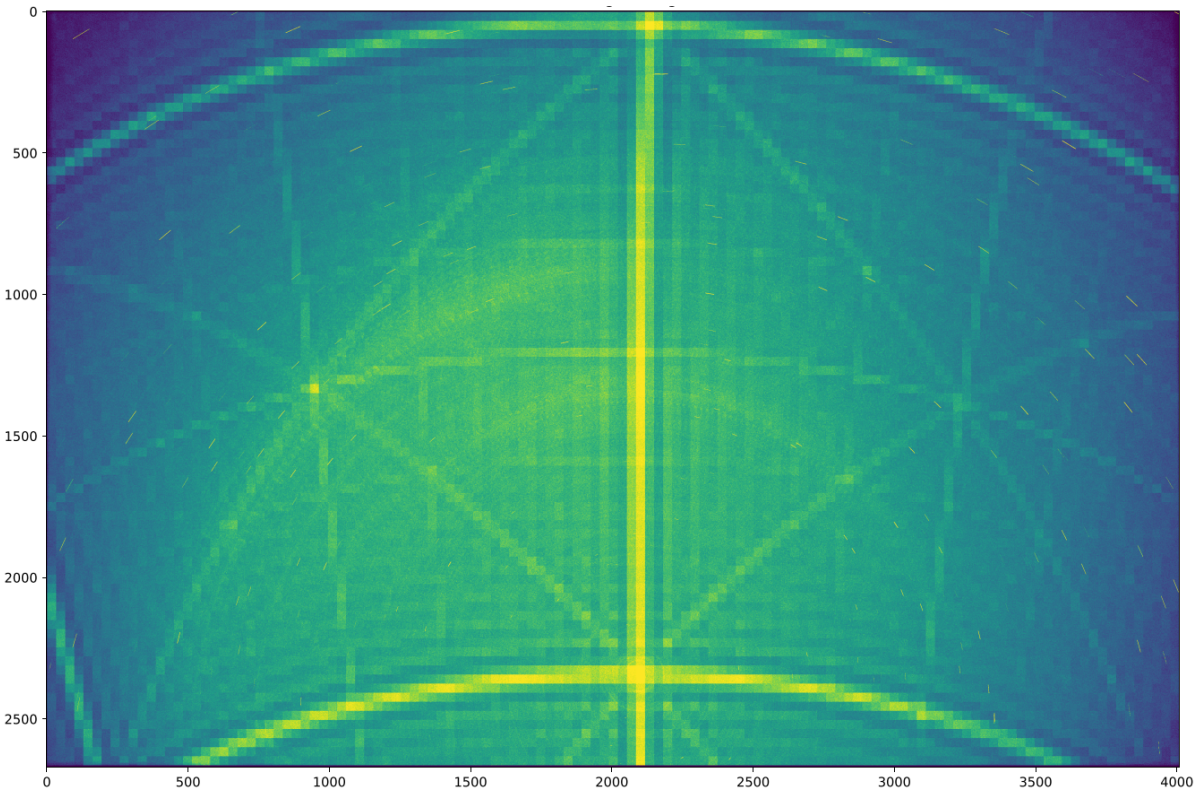


Figure A.8: LSS All Stack Background image

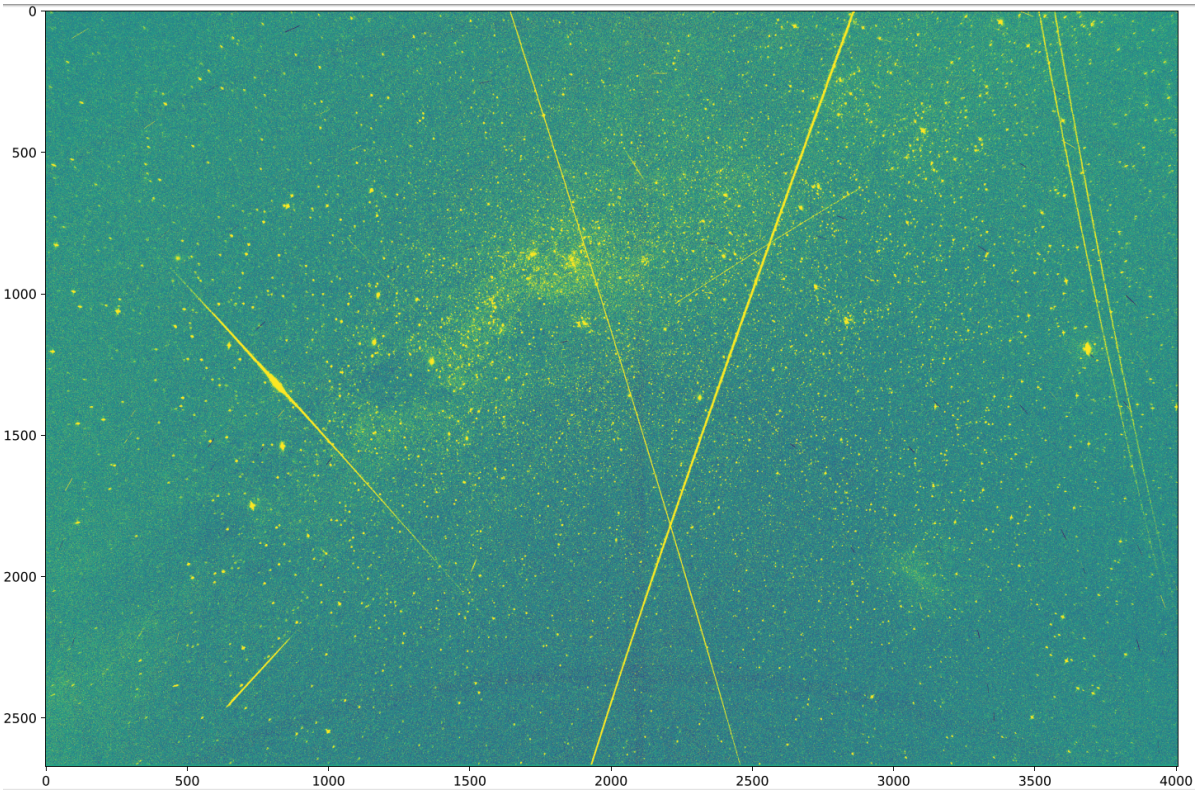


Figure A.9: LSS84 All Stack Background subtracted image, which follows from subtracting fig. A.8 from fig. A.3.

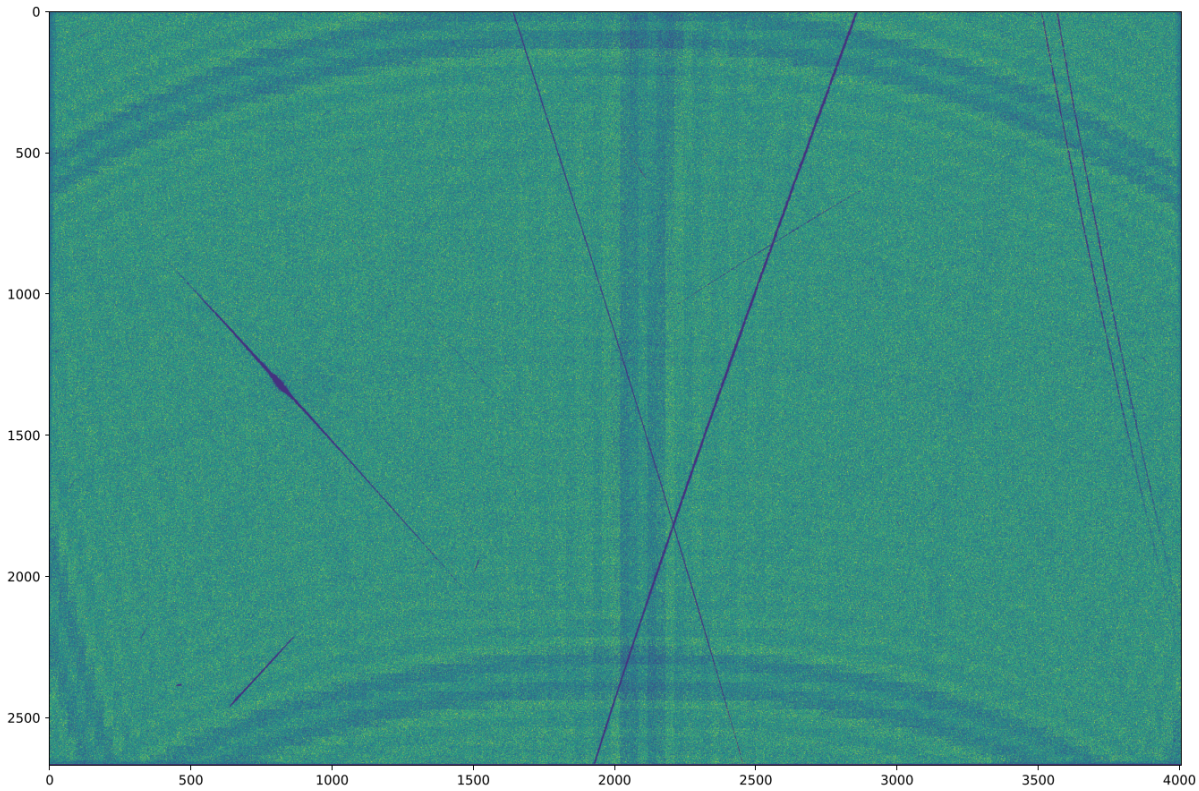


Figure A.10: LSS84 index difference filter response

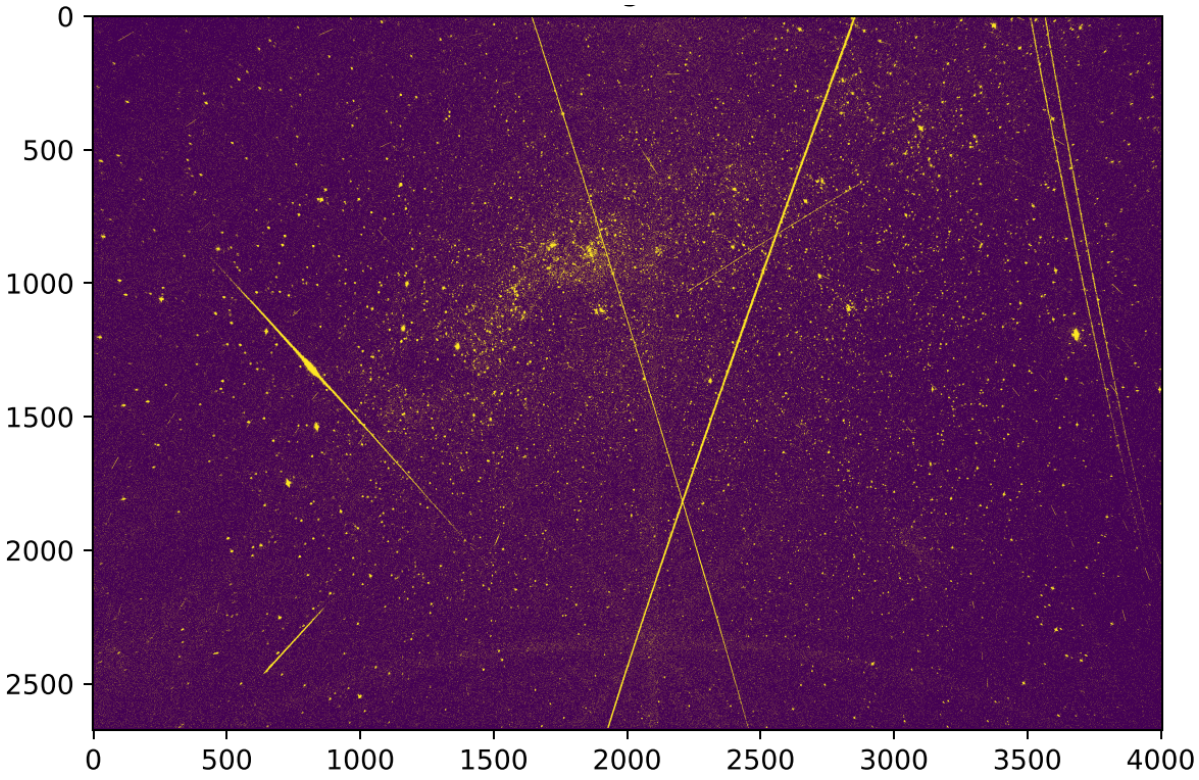


Figure A.11: LSS84 All Stack Background subtracted binary image for $\sigma = 2$

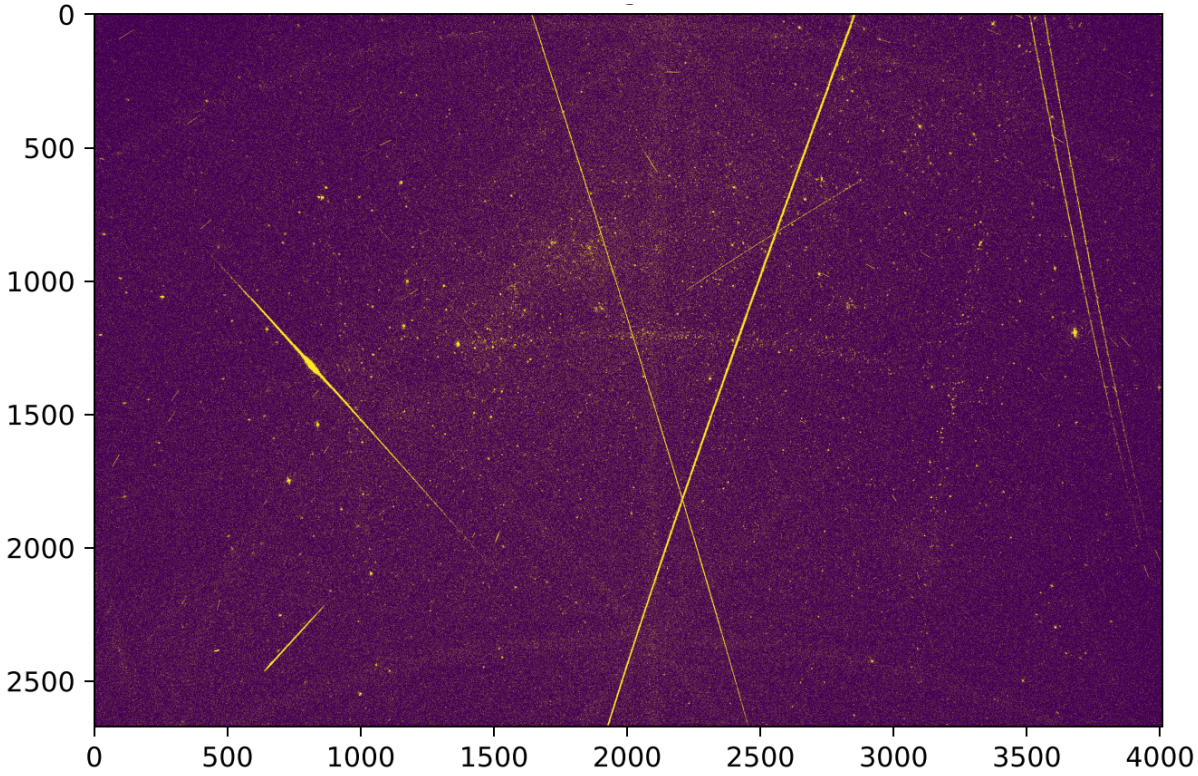


Figure A.12: LSS84 Even odd difference binary image for $\sigma = 2$

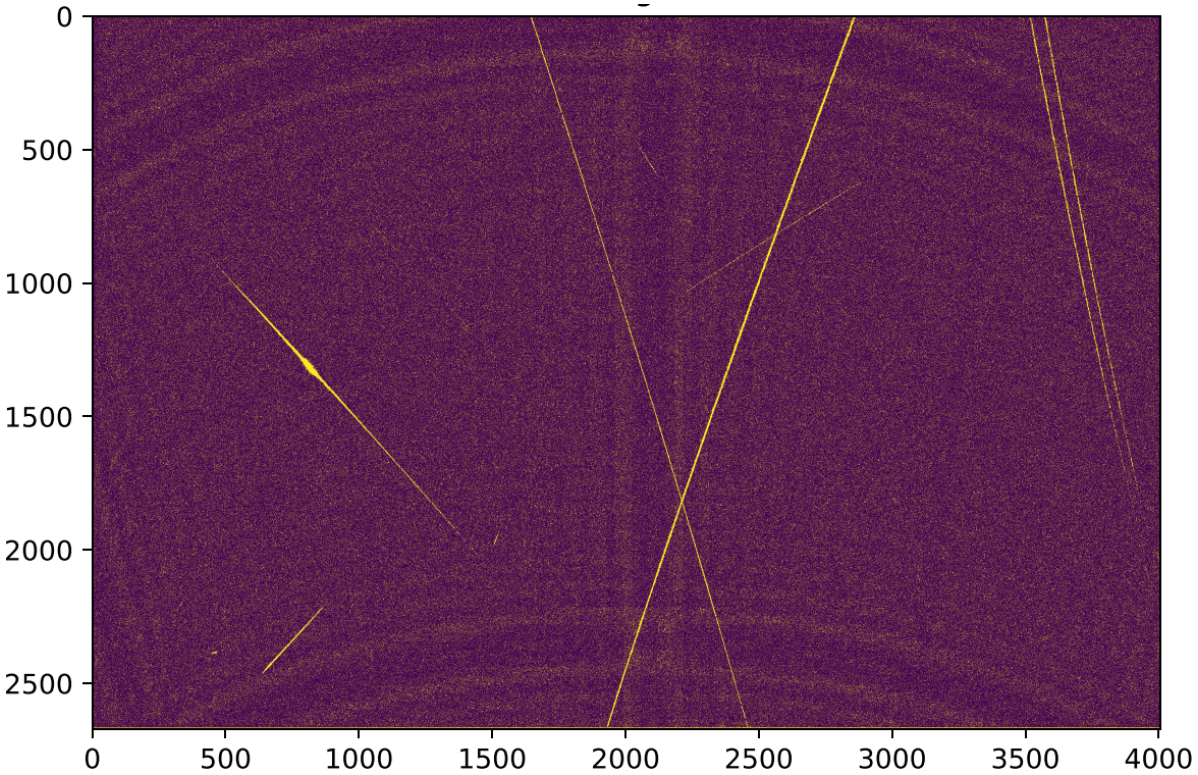


Figure A.13: LSS84 index difference filter binary image for $\sigma = 1.5$

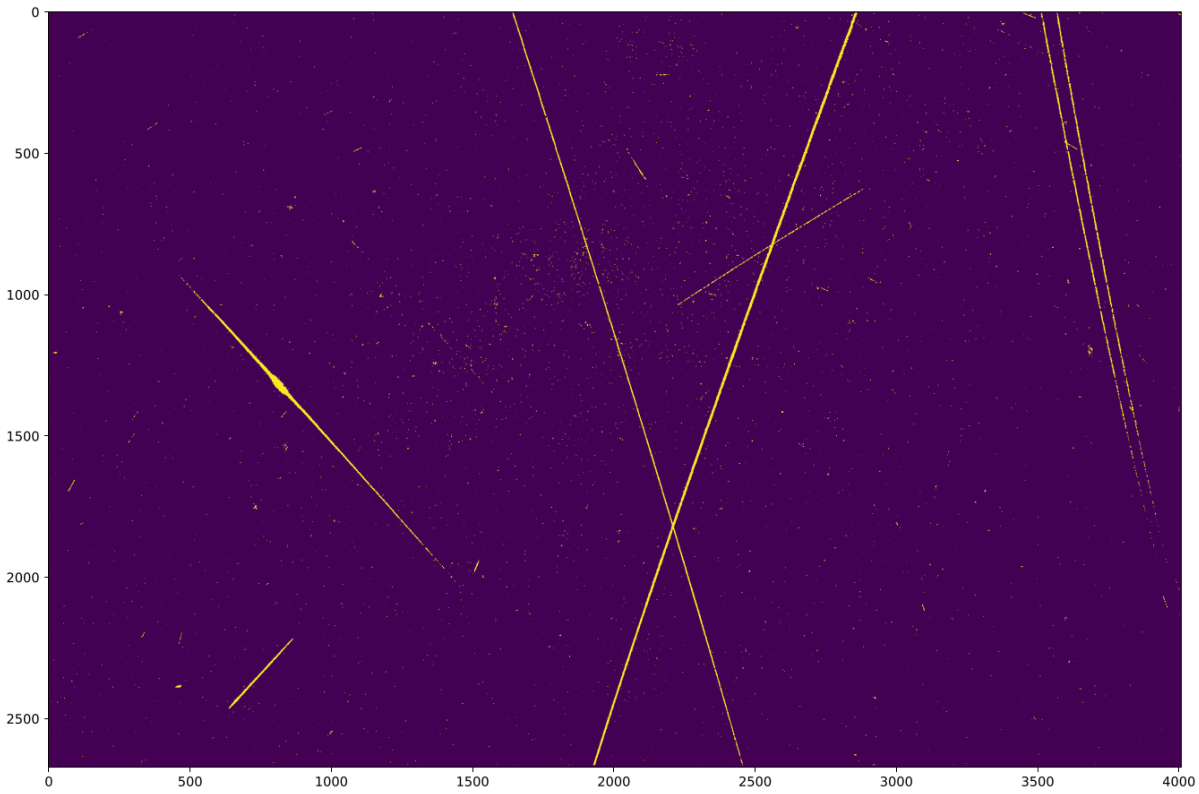


Figure A.14: LSS84 Binary propagation result which is the multiplication of fig. A.11, fig. A.11 and fig. A.11. Since the moon was not present (above the horizon), the erosion parameter was set to 1.

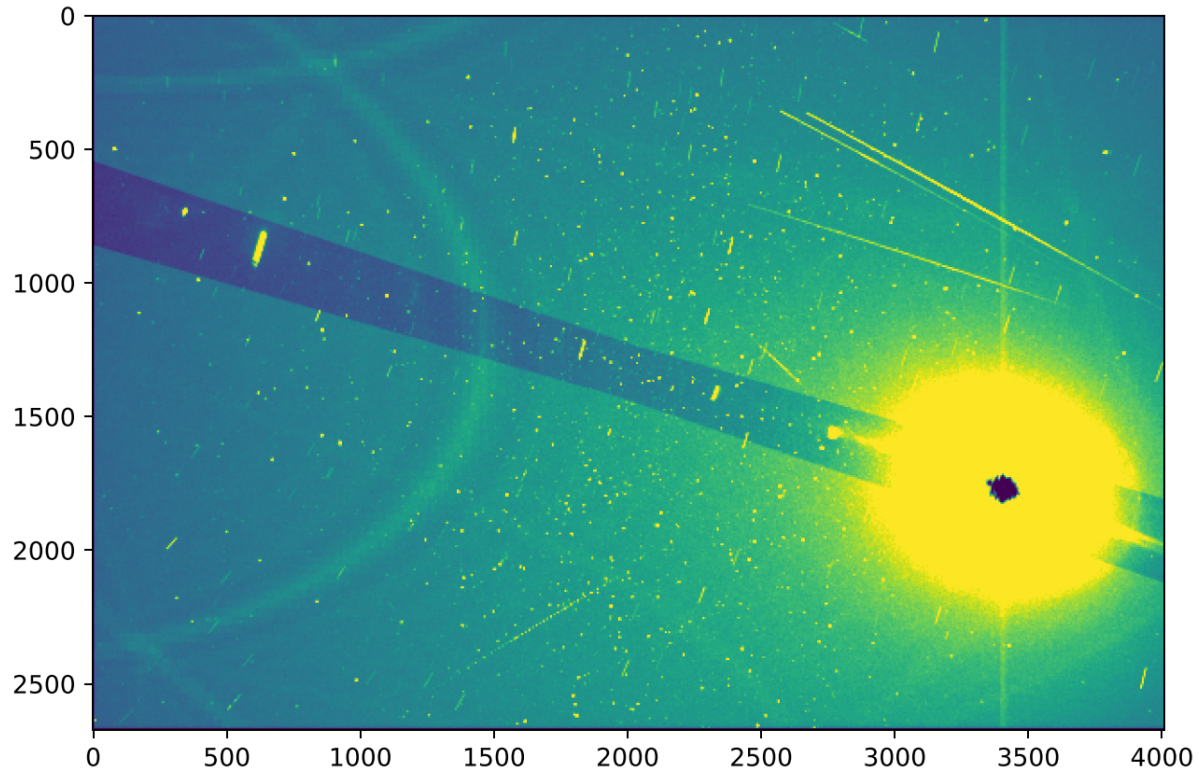


Figure A.15: LSW20 value stack with a bright present moon. The mask is also highlighted by subtracting one standard deviation from the image within the mask.

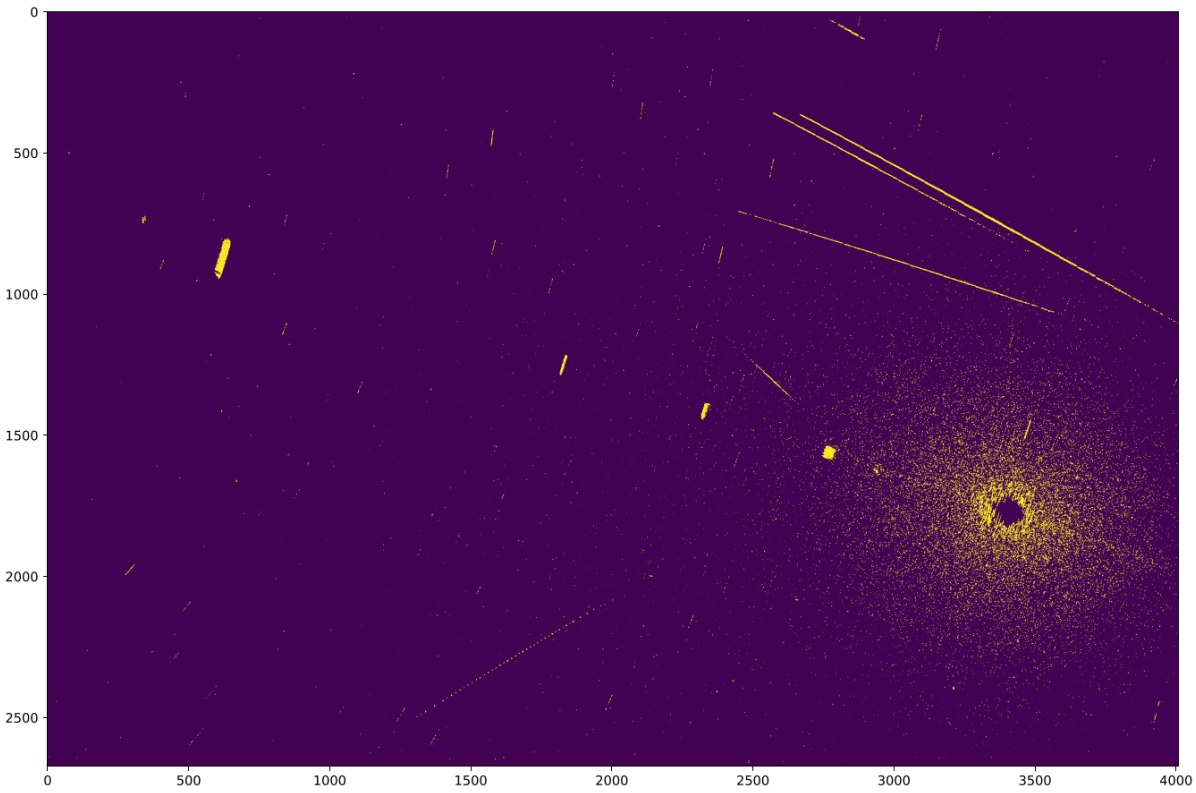


Figure A.16: LSW20 detection image without the masking and erosion parameter set to 1.

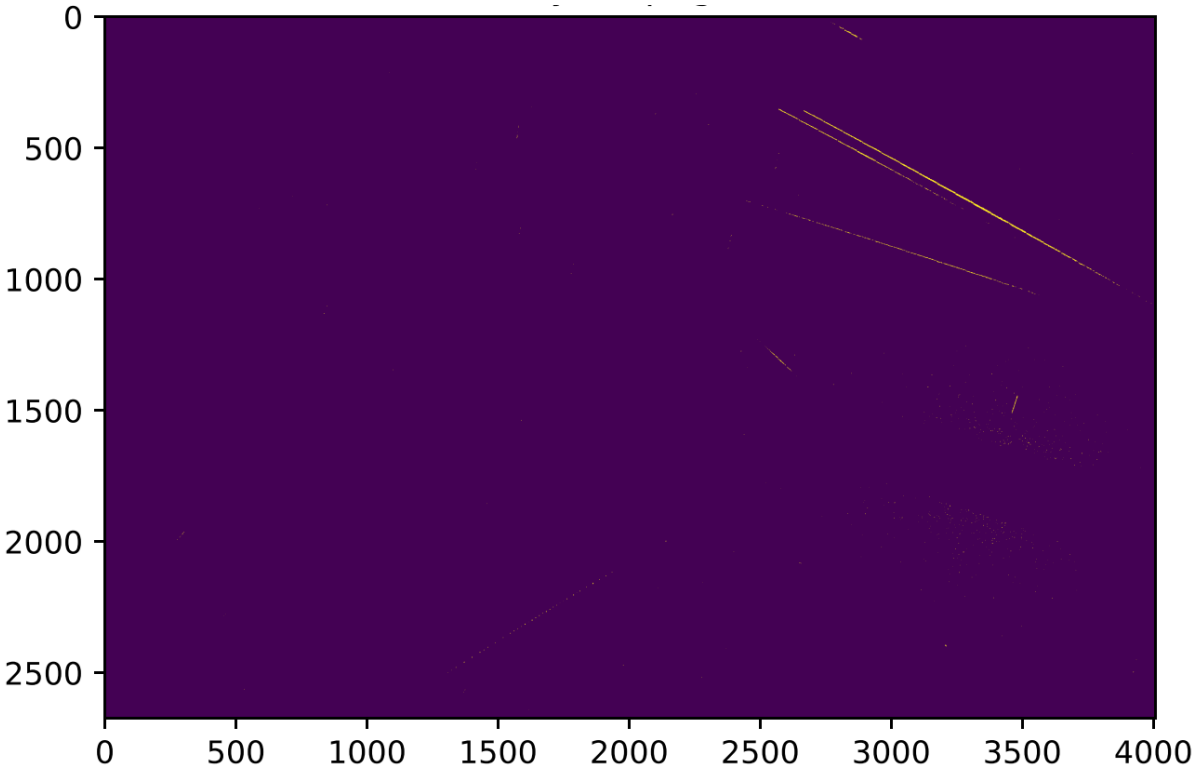


Figure A.17: LSW20 detection image with the masking and erosion parameter set to 3.

B

Pipeline design

This serves as a final overview of the Python scripts that are written to go from the input images to endpoints for the detected objects. For each script the inputs, outputs and run-time on the Leiden server are stated. A flowchart will show the execution order and the connections between different scripts.

B.1. filterTLE

The first script that needs to be ran is one that can filter out objects from the TLE catalog. This can be based on altitude or any other info of a TLE. This script also creates the groups that were used in the performance estimations. A new category of objects can easily be made to gauge the performance of the method.

- Inputs: Optional altitude limit (or another parameter of the TLE)
- Outputs: Filtered TLE catalog for future processing, groups of TLE's for performance estimation
- Run-time: 1 second
- Optimizations: none

B.2. createBlacklist

The next optional script that can be run is to reduce the input data that needs to be processed based on what was investigated in chapter 3. By giving an altitude limit the script returns the observations windows per camera that are not of main interest. This list is then imported by the next script to ignore certain input images, improving the efficiency of the pipeline as it can remove up to 50 % of the input data. As was shown in table 3.2, the run-time depends on the discretisation quality of the simulation. The results converged after about $150,000$ points.

- Inputs: Altitude limit, Station info, date
- Outputs: Blacklist with filenames that can be ignored in the next steps
- Run-time: 9 minutes for 0.5 degree angular resolution
- Optimizations: can be executed at lower resolutions for little time gain.

B.3. createStacks

This script opens all the available images that are taken from the night before as well as the AS to create the stacked images. Before the images are stacked, the plain images are calibrated with the master dark frame that is made from the available dark frames. When creating the stack, the filtered TLE catalog is also processed with the SGP4 method to determine the position of each object with

respect to the star background. For each stack there are 50 images available and thus 51 timestamps (starts and ends). For each object the timestamps are evaluated, and if the object is in view the sky positions are returned to a dictionary. This script can be run in parallel as it is done per camera.

- Inputs: Blacklist, filtered TLE catalog, input images (science + calibration)
- Outputs: Stacks, dictionary with objects that are in view per stack
- Run-time: the creation of one stack takes 200 seconds, the calculation of all the satellite positions takes 330 seconds
- Optimizations: Not exporting plots and intermediate products, exporting the *pdf* and *fits* files takes 80 seconds in total.

B.4. createCalibrationFrames

This script takes the available calibration frames for the night and creates master calibration frames.

- Inputs: calibration images
- Outputs: master calibration frames
- Run-time: 1 minute 30 seconds per frame, per camera
- Optimizations: Only execute for calibration frames that are used.

B.5. imageProcessing

This script takes the created stacks and processes them into the detection images, this was described in chapter 4. Like the previous script it can be run in parallel.

- Inputs: Stacked images
- Outputs: Detection images
- Run-time: the operations take 10 seconds per stack/image
- Optimizations: Not exporting plots and intermediate products, exporting the *pdf* and *fits* files takes 100 seconds in total.

B.6. featureAnalysis

This script goes through all the data points from the dictionary that was created by the *createStacks* script. It analyses the track shape of all the objects that are in view during the night for one camera. Besides outputting some plots it gives an indication of the straightness of the line segments. These results are described in chapter 5 and are used in the detection and endpoint determination script.

- Inputs: Dictionary with satellite positions per stack
- Outputs: metrics to optimize the detection of features
- Run-time: 5 minutes per camera (but depends on number of timestamps and TLE objects)
- Optimizations: In theory only has to be ran once to obtain the parameters.

B.7. trackDetection

This is the final script that does the detection and endpoint determination. It takes the detection images and performs the routines described in chapter 6, after which the detected and matched tracks are evaluated with the two endpoint determination methods. Also this script can be run in parallel per camera.

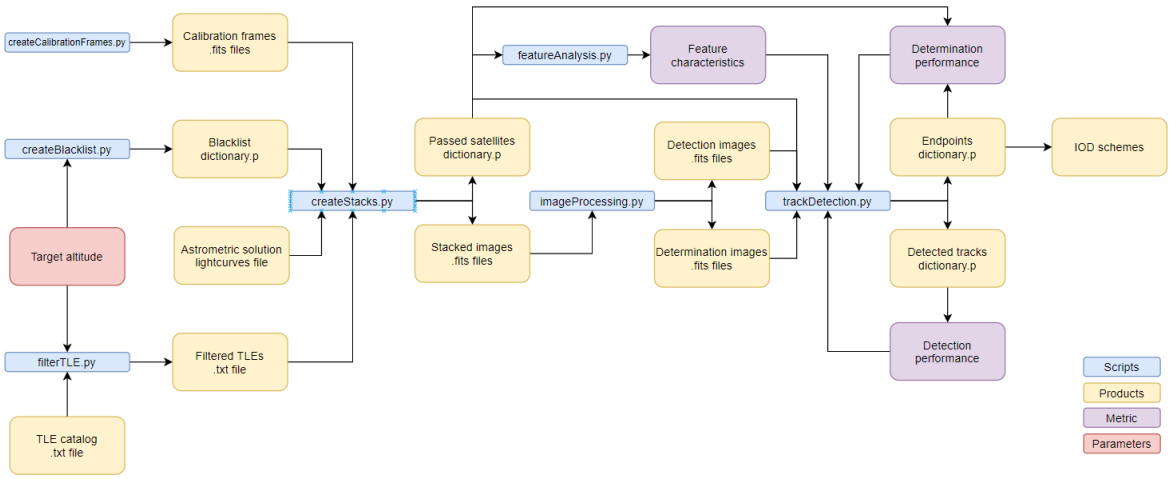


Figure B.1: The written scripts and connections between them, the legend shows what type of blocks are available. Basically the only parameters

- Inputs: Dictionary with passed satellites, detection images, filtered TLE catalog, results from feature analysis
- Outputs: Endpoints for detected streaks in the images
- Run-time: This depends on the number of detections per stack. Detection and track isolation takes about 10 seconds in total. Then per detection the satellite matching takes 2 seconds at most. The endpoint determination methods differ, the double index prediction takes 10 seconds and the polyfit index trace takes 30 seconds (on average).
- Optimizations: only running the endpoint determination for the best performing method and regressor.

B.8. detectionAnalysis

This script evaluates the performance of the detections, and was used to compare the detections from the FOTOS1 method and the newly developed methods against each other and the SGP4 positions. It is not actually a standalone script, but rather embedded in the *trackDetection* script. This is quicker as all data is already loaded and does not need to be stored and opened again in the meantime. It is possible and likely that this script will be a stand-alone execution at a later stage.

- Inputs: Dictionary with passed satellites, filtered TLE catalog, endpoints per identified object
- Outputs: Performance metrics
- Run-time: 5 minutes
- Optimizations: none besides not running it.

B.9. Overview of the scripts and outputs

As can be seen in fig. B.1, the scripts execution order follows naturally and matches with what is described in the body of the thesis and the order above. Basically the only parameter is the optional limiting altitude, where all other parameters can be tweaked based on metrics from the analysis scripts.

B.10. Time estimation

The total time estimate depends mostly on the duration of the night as that determines how many stacks are present. However, the number of stacks does not directly correspond to the total time needed to process the images. As there are:

Table B.1: Scripts and functions that need to be executed for every stack, for this night of testdata there were 96 stacks.

Script or function	Time per stack [s]	Total time [hrs]
createStacks.py	200	5.33
TLE calculation	330	8.80
Image operations	10	0.27
Track detection	2	0.05
Track isolation	1	0.03
Total	543	14.48

Table B.2: Functions that need to be executed for every detected track. On average there were 5 track detections per stack, and 96 stacks in this night of testdata.

Script or function	Time per stack [s]	Total time [hrs]
Satellite matching	2	0.27
DIP 3x	10	1.33
PIT 3x	30	4.00
Total	42	5.60

- Scripts that need to be run once for initialization
- Scripts that need to be run once per camera
- Scripts or functions that need to be run per stack of images
- Scripts or functions that need to be run for every detected track in one stack

For the night of testdata we make an estimate for the total time needed to process all the data from one camera.

The scripts that always need to be ran in prior each night are *filterTLE.py* and *createBlacklist.py*. The total execution time for these scripts in total is approximately 9 minutes and is for 99% due to the *createBlacklist.py* script.

After that we run the *createCalibrationFrames.py* script for each camera, which takes 90 seconds per calibration frame. And since only the master dark calibration frame is used, this is also the total per camera.

Then for creating the detection images we need to create the stacks and perform the image operations to obtain the detection image. To do this we run the scripts *createStacks.py* and *imageProcessing.py* Another operation that is performed during each stack is the determination of the satellite positions from the catalog (within *createStacks.py*). What is also done for each stack are the **Detection** and **Isolation** functions in the *trackDetection.py* script. These functions take about 3 seconds in total. The total time per stack and camera for the night of testdata is approximated in table B.1.

The majority of the execution time per stack comes from the TLE calculation and the creation of the difference images and stacks. For 96 stacks in the testdata the total processing time per stack is approximately 9 minutes.

After the detection step there are functions that need to be run in *trackDetection.py* script for every isolated track. The track needs to be checked against the catalog of satellite with the **SatelliteMatching** function, after which the endpoints are determined with **DoubleIndexPredict** and **PolyfitIndexTrace**. On average there are about 5 detections per image, this is estimated from table 6.4. The total number of tracks per camera is then approximately 480. The total time per track and camera for the night of testdata is approximated in table B.2.

The total time per camera then totals to 20.5 hours. This estimate still includes the two different endpoint prediction methods and regressors. Thus the total can still be reduced to about 17 hours. Note that the number of detected tracks per stack depends on the duration of the night. The majority of track detections happen at dawn and dusk, thus for longer nights the average will be lower than 5 tracks per stack.

The testdata is for a fairly short night in summer and it thus seems that the application of the data reduction simulation will be needed for longer nights. The application of the reduction simulation also

depends on if the pipeline can run during observation. If this is not possible, processing the images during the day is infeasible when days become very short (11 hours). Not only due to the increased amount of data, but also since there is less time to process.

Dependent on the PC hardware of the station, the run-times need to be re-evaluated such that it can be decided if an altitude limit should be enforced to reduce the overall computation time.

Note that this is the current pipeline and not the one for final use. There are still several steps missing in this pipeline, like the classification and the orbit determination.

Bibliography

[1] Richard Berry and James Burnell. *The handbook of astronomical image processing*. Willmann-Bell, Richmond, VA, 2nd ed edition, 2005. ISBN 978-0-943396-82-8.

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] Thomas A Caswell, Michael Droettboom, Antony Lee, Elliott Sales De Andrade, John Hunter, Tim Hoffmann, Eric Firing, Jody Klymak, David Stansby, Nelle Varoquaux, Jens Hedegaard Nielsen, Benjamin Root, Ryan May, Phil Elson, Jouni K. Seppänen, Darren Dale, Jae-Joon Lee, Damon McDougall, Andrew Straw, Paul Hobson, Christoph Gohlke, Hannah, Tony S Yu, Eric Ma, Adrien F. Vincent, Steven Silvester, Charlie Moad, Nikita Kniazev, Elan Ernest, and Paul Ivanov. matplotlib/matplotlib: REL: v3.4.2, May 2021. URL https://zenodo.org/record/4743323.

[4] The Astropy Collaboration, A. M. Price-Whelan, B. M. Sipőcz, H. M. Günther, P. L. Lim, S. M. Crawford, S. Conseil, D. L. Shupe, M. W. Craig, N. Dencheva, A. Ginsburg, J. T. VanderPlas, L. D. Bradley, D. Pérez-Suárez, M. de Val-Borro, T. L. Aldcroft, K. L. Cruz, T. P. Robitaille, E. J. Tollerud, C. Ardelean, T. Babej, M. Bachetti, A. V. Bakanov, S. P. Bamford, G. Barentsen, P. Barmby, A. Baumbach, K. L. Berry, F. Biscani, M. Boquien, K. A. Bostroem, L. G. Bouma, G. B. Brammer, E. M. Bray, H. Breytenbach, H. Buddelmeijer, D. J. Burke, G. Calderone, J. L. Cano Rodríguez, M. Cara, J. V. M. Cardoso, S. Cheedella, Y. Copin, D. Crichton, D. DÁvella, C. Deil, É Depagne, J. P. Dietrich, A. Donath, M. Droettboom, N. Earl, T. Erben, S. Fabbro, L. A. Ferreira, T. Finethy, R. T. Fox, L. H. Garrison, S. L. J. Gibbons, D. A. Goldstein, R. Gommers, J. P. Greco, P. Greenfield, A. M. Groener, F. Grollier, A. Hagen, P. Hirst, D. Homeier, A. J. Horton, G. Hosseinzadeh, L. Hu, J. S. Hunkeler, Ž Ivezić, A. Jain, T. Jenness, G. Kanarek, S. Kendrew, N. S. Kern, W. E. Kerzendorf, A. Khvalko, J. King, D. Kirkby, A. M. Kulkarni, A. Kumar, A. Lee, D. Lenz, S. P. Littlefair, Z. Ma, D. M. Macleod, M. Mastropietro, C. McCully, S. Montagnac, B. M. Morris, M. Mueller, S. J. Mumford, D. Muna, N. A. Murphy, S. Nelson, G. H. Nguyen, J. P. Ninan, M. Nöthe, S. Ogaz, S. Oh, J. K. Parejko, N. Parley, S. Pascual, R. Patil, A. A. Patil, A. L. Plunkett, J. X. Prochaska, T. Rastogi, V. Reddy Janga, J. Sabater, P. Sakurikar, M. Seifert, L. E. Sherbert, H. Sherwood-Taylor, A. Y. Shih, J. Sick, M. T. Silbiger, S. Singanamalla, L. P. Singer, P. H. Sladen, K. A. Sooley, S. Sornarajah, O. Streicher, P. Teuben, S. W. Thomas, G. R. Tremblay, J. E. H. Turner, V. Terrón, M. H. van Kerkwijk, A. de la Vega, L. L. Watkins, B. A. Weaver, J. B. Whitmore, J. Woillez, and V. Zabalza. The Astropy Project: Building an inclusive, open-science project and status of the v2.0 core package. *The Astronomical Journal*, 156(3):123, August 2018. ISSN 1538-3881. doi: 10.3847/1538-3881/aabc4f. URL http://arxiv.org/abs/1801.02634. arXiv: 1801.02634 Citation Key: astropy.

[5] M Deserno. How to generate equidistributed points on the surface of a sphere. page 1, September 2004. URL https://www.cmu.edu/biolphys/deserno/pdf/sphere_equi.pdf.

[6] Wei Dong and Zhao Chang-yin. An Accuracy Analysis of the SGP4/SDP4 Model. *Chinese Astronomy and Astrophysics*, 34(1):69–76, January 2010. ISSN 02751062. doi: 10.1016/j.chinastron.2009.12.009. URL https://linkinghub.elsevier.com/retrieve/pii/S0275106209001404.

[7] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Pearson, New York, NY, fourth edition, global edition edition, 2018. ISBN 978-1-292-22304-9. OCLC: 1044698635, Citation Key: DIPGonzalez.

[8] Olivier R. Hainaut and Andrew P. Williams. Impact of satellite constellations on astronomical observations with ESO telescopes in the visible and infrared domains. *Astronomy & Astrophysics*, 636:A121, April 2020. ISSN 0004-6361, 1432-0746. doi: 10.1051/0004-6361/202037501. URL https://www.aanda.org/10.1051/0004-6361/202037501.

[9] M.A. Hapgood. Space physics coordinate transformations: A user guide. *Planetary and Space Science*, 40(5):711–717, May 1992. ISSN 00320633. doi: 10.1016/0032-0633(92)90012-D. URL https://linkinghub.elsevier.com/retrieve/pii/003206339290012D.

[10] P. Hickson. A fast algorithm for the detection of faint orbital debris tracks in optical images. *Advances in Space Research*, 62(11):3078–3085, December 2018. ISSN 02731177. doi: 10.1016/j.asr.2018.08.039. URL http://arxiv.org/abs/1809.00239. arXiv: 1809.00239.

[11] Hough, P.V.C. METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS, December 1962.

[12] Donald J. Kessler and Burton G. Cour-Palais. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research*, 83(A6):2637, 1978. ISSN 0148-0227. doi: 10.1029/JA083iA06p02637. URL http://doi.wiley.com/10.1029/JA083iA06p02637.

[13] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: a LLVM-based Python JIT compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15*, pages 1–6, Austin, Texas, 2015. ACM Press. ISBN 978-1-4503-4005-2. doi: 10.1145/2833157.2833162. URL http://dl.acm.org/citation.cfm?doid=2833157.2833162.

[14] A.-L. Lesage, J. F. P. Spronck, R. Stuik, F. Bettonvil, D. Pollaco, and I. A. G. Snellen. MASCARA: the multi-site all-sky CAameRA: concept and first results. page 914514, Montréal, Quebec, Canada, July 2014. doi: 10.1117/12.2055997. URL http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2055997.

[15] Jack J. Lissauer and Imke de Pater. *Fundamental Planetary Science: Physics, Chemistry and Habitability*. Cambridge University Press, 1 edition, September 2013. ISBN 978-0-521-85330-9 978-1-139-05046-3 978-0-521-61855-7. doi: 10.1017/CBO9781139050463. URL https://www.cambridge.org/highereducation/books/fundamental-planetary-science/7D60400975EFC2A392B55EA8F3BB2233#contents.

[16] Martin P Lévesque. *Image processing technique for automatic detection of satellite streaks*. 2007.

[17] J. Matas, C. Galambos, and J. Kittler. Progressive Probabilistic Hough Transform. In *Procedings of the British Machine Vision Conference 1998*, pages 26.1–26.10, Southampton, 1998. British Machine Vision Association. ISBN 978-1-901725-04-9. doi: 10.5244/C.12.26. URL http://www.bmva.org/bmvc/1998/papers/d176/h176.htm.

[18] Guy Nir, Barak Zackay, and Eran O. Ofek. Optimal and Efficient Streak Detection in Astronomical Images. *The Astronomical Journal*, 156(5):229, October 2018. ISSN 1538-3881. doi: 10.3847/1538-3881/aaddff. URL https://iopscience.iop.org/article/10.3847/1538-3881/aaddff.

[19] Florin Oniga, Melania Miron, Radu Danescu, and Sergiu Nedevschi. Automatic recognition of low earth orbit objects from image sequences. In *2011 IEEE 7th International Conference on Intelligent Computer Communication and Processing*, pages 335–338, Cluj-Napoca, Romania, August 2011. IEEE. ISBN 978-1-4577-1479-5. doi: 10.1109/ICCP.2011.6047892. URL http://ieeexplore.ieee.org/document/6047892/.

[20] Jang-Hyun Park, Hong-Suh Yim, Young-Jun Choi, Jung Hyun Jo, Hong-Kyu Moon, Young-Sik Park, Young-Ho Bae, Sun-Youp Park, Dong-Goo Roh, Sungki Cho, Eun-Jung Choi, Myung-Jin Kim, and Jin Choi. OWL-NET: A GLOBAL NETWORK OF ROBOTIC TELESCOPES DEDICATED TO SSA OBSERVATION. page 5.

[21] Rhorom Priyatikanto, Bahar Religia, Abdul Rachman, and Tiar Dani. Towards photometry pipeline of the Indonesian space surveillance system. page 050011, Bandung, Indonesia, 2015. doi: 10.1063/1.4930672. URL http://aip.scitation.org/doi/abs/10.1063/1.4930672.

[22] Johann Radon. On the determination of functions from their integral values along certain manifolds. *IEEE Transactions on Medical Imaging*, 5(4):170–176, December 1986. ISSN 0278-0062, 1558-254X. doi: 10.1109/TMI.1986.4307775. URL http://ieeexplore.ieee.org/document/4307775/.

[23] Radu Danescu and Anca Ciurte. Automatic Detection of MEO Satellite Streaks from Single Long Exposure Astronomic Images:. In *Proceedings of the 9th International Conference on Computer Vision Theory and Applications*, pages 538–544, Lisbon, Portugal, 2014. SCITEPRESS - Science and and Technology Publications. ISBN 978-989-758-003-1 978-989-758-004-8 978-989-758-009-3. doi: 10.5220/0004721505380544. URL http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0004721505380544.

[24] Brandon Craig Rhodes. PyEphem: Astronomical Ephemeris for Python, December 2011. Pages: ascl:1112.014 _eprint: 1112.014.

[25] F. Samadzadegan and F. Alidoost. THE DESIGN AND IMPLEMENTATION OF AN OPTICAL ASTRONOMICAL SATELLITE TRACKING SYSTEM. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XL-1/W3:25–30, September 2013. ISSN 2194-9034. doi: 10.5194/isprsarchives-XL-1-W3-25-2013. URL http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XL-1-W3/25/2013/.

[26] SciPy 1.0 Contributors, Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020. ISSN 1548-7091, 1548-7105. doi: 10.1038/s41592-019-0686-2. URL http://www.nature.com/articles/s41592-019-0686-2.

[27] R. Stuik, J. I. Bailey III, P. Dorval, G. J. J. Talens, I. Laginja, S. N. Mellon, B. B. D. Lomberg, S. M. Crawford, M. J. Ireland, E. E. Mamajek, and M. A. Kenworthy. bRing: An observatory dedicated to monitoring the β Pictoris b Hill sphere transit. *Astronomy & Astrophysics*, 607: A45, November 2017. ISSN 0004-6361, 1432-0746. doi: 10.1051/0004-6361/201731679. URL http://arxiv.org/abs/1709.01325. arXiv: 1709.01325.

[28] G. J. J. Talens, J. F. P. Spronck, A.-L. Lesage, G. P. P. L. Otten, R. Stuik, D. Pollacco, and I. A. G. Snellen. The Multi-site All-Sky CAmeRA (MASCARA): Finding transiting exoplanets around bright ($m_v < 8$) stars. *Astronomy & Astrophysics*, 601:A11, May 2017. ISSN 0004-6361, 1432-0746. doi: 10.1051/0004-6361/201630319. URL http://www.aanda.org/10.1051/0004-6361/201630319.

[29] G. J. J. Talens, E. R. Deul, R. Stuik, O. Burggraaff, A.-L. Lesage, J. F. P. Spronck, S. N. Mellon, J. I. Bailey, E. E. Mamajek, M. A. Kenworthy, and I. A. G. Snellen. Data calibration for the MASCARA and bRing instruments. *Astronomy & Astrophysics*, 619:A154, November 2018. ISSN 0004-6361, 1432-0746. doi: 10.1051/0004-6361/201834070. URL https://www.aanda.org/10.1051/0004-6361/201834070.

[30] David Vallado and Paul Crawford. SGP4 Orbit Determination. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Honolulu, Hawaii, August 2008. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-001-7. doi: 10.2514/6.2008-6770. URL https://arc.aiaa.org/doi/10.2514/6.2008-6770.

[31] David A Vallado and Paul J Cefola. IAC-12-A6.6.11 TWO-LINE ELEMENT SETS – PRACTICE AND USE. page 15.

[32] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, June 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. URL https://peerj.com/articles/453.

[33] M van Ginkel, C L Luengo Hendriks, and L J van Vliet. A short introduction to the Radon and Hough transforms and how they relate to each other. page 11.

[34] James Richard Wertz. *Mission geometry: orbit and constellation design and management: spacecraft orbit and attitude systems*. Number 13 in Space technology library. Microcosm Press : Kluwer Academic Publishers, El Segundo, Calif. : Dorfrecht ; Boston, 2001. ISBN 978-0-7923-7148-9.

[35] Thomas Wijnen, Remko Stuik, Michiel Rodenhuis, and Marco Langbroek. Eindrapport FOTOS, July 2018.

[36] W.N.J. Rood. *Automated Satellite Track Detection and Endpoint Determination in Astronomical Images*. MSc Thesis Literature Study, Delft University of Technology, Delft, October 2020. URL https://github.com/willemrood/litstudy.