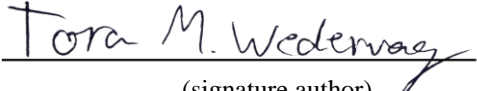




Universitetet
i Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study program/specialization: Automation and Signal Processing	Spring semester, 2016 Open/ Confidential
Author: Tora Marie Wedervang	 (signature author)
Instructor: Kjersti Engan Supervisors: Vegard Arneson Lars Erling Bråten	
Title of Master's Thesis: Video Analysis in Land Mobile Satellite Communication Norwegian title: Videoanalyse i Landmobil Satellittkommunikasjon	
ECTS:30	
Subject headings: Satellite communication Hemispherical lens Image processing Segmentation	Pages: 46 + attachments/other: 2 Stavanger, 15/06/2016 Date/year

Abstract

Satellite communication is exposed to big variations in signal strength due to attenuating or blocking objects in the signal path. This is especially apparent in mobile satellite communication where the antenna is constantly on the move. If a hemispherical video is captured along with the satellite data, a better understanding of what is blocking or attenuating the signal can be achieved. The video can be manually segmented into attenuating objects (vegetation) and blocking objects (buildings or terrain), or an automated system can be made to segment the video automatically. This thesis proposes such a system for segmenting a hemispherical video into line of sight and non-line of sight segments. These are then used to propose a statistical model of probable signal strength at different azimuth and elevation angles in relation to the vehicle and in relation to cardinal directions.

The thesis and proposed system are based on earlier works. The main advancement in this thesis is the improvement of computation time and the system's use in building a statistical model of probabilities of blockage as a function of elevation and azimuth angles in various environments. This can then be used in further analysis and system requirements of future satellite communications on the move and communications with non-geostationary satellites.

Preface

This master thesis is written as a part of the Automation and Signal Processing program at the University of Stavanger (UiS). It was given as an assignment from the Norwegian Defence Research Establishment (FFI).

My supervisors during the work with the thesis were Vegard Arneson and Lars-Erling Bråten from FFI and Kjersti Engan from UiS. They have all been a big help; Arneson and Bråten with technical input on the satellite communication aspects, and Engan with input on the image and video processing. All technical input has been greatly appreciated, but I would also like to extend a special thanks for the motivation and inspiration as to which methods might work when I got stuck on a problem. Thank you for helping me in getting this thesis to where it is today.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	The objectives of this thesis	2
1.2.1	Problem Formulation	3
1.2.2	Limitations	3
1.3	Background	4
1.3.1	Background Material	4
1.3.2	Previous Works	4
1.3.3	Data sets	6
1.3.4	Encoding of video format	7
2	Theory	8
2.1	Image processing	8
2.1.1	Otsu’s thresholding method	8
2.1.2	Optical Character Recognition	9
2.1.3	Hough transform	10
2.1.4	Statistical Region Merging	11
3	Verification of Baseline Work	13
3.1	Bergerskogen’s work	13
3.1.1	Limitations	14
3.2	Verification	15
4	Proposed system	18
4.1	Initiation block	18
4.1.1	Input	18
4.1.2	Finding environment type	19
4.1.3	Weighting of color components	19
4.1.4	Polar conversion variables	19
4.2	Main blocks	20
4.2.1	Polar conversion	20
4.2.2	Segmentation	22
4.2.3	Optimization	22
4.3	End blocks	23
4.3.1	Save to Video	23
4.3.2	Save to Condition Matrix	23

5	Experiment and Results	25
5.1	Preparatory work	25
5.2	Weighting of color components	26
5.3	Finding threshold	27
5.4	Statistical Region Merging	28
5.5	Optimization	29
	5.5.1 Finding segment area	30
	5.5.2 Choice of Increment size	32
	5.5.3 Dropping frames	33
5.6	Three states	34
	5.6.1 Statistical Region Merging	34
	5.6.2 Houghlines	35
	5.6.3 Otsu's thresholding	36
5.7	Statistical Model	37
6	Discussion	40
6.1	Limitations	40
	6.1.1 Accuracy measurement	40
6.2	Objective achievement	41
6.3	Computation Time	41
6.4	Future possibilities	42
	6.4.1 Improving the current system	42
	6.4.2 Making the system more applicable	42
	6.4.3 Motion tracking	43
	6.4.4 Three states	43
7	Conclusion	46
A	Appendices	48
A.1	Original Problem formulation	48
A.2	Segmented video with overlay	48
A.3	Video showing delta areas for block and dilation	48
A.4	MATLAB code for Video analyzer .zip	48

List of Figures

1.1	Driving route in measuring campaign	2
1.2	Setup of vehicle in measuring campaign. The image is taken from the MiLADY website[1]	5
1.3	Example frames from different videos	6
2.1	OCR with Matrix matching (blue) and Feature extraction (green)	9
2.2	Hough transform applied to video frame with buildings	10
2.3	Statistical Region Merging on sunny frame	12
3.1	Block diagram of Bergerskogen’s algorithm	13
3.2	Examples of different kinds of environment	15
3.3	Selected stills from the visual verification videos. Top: summeropen, Middle: mountainsun, Bottom: rainbuild	17
4.1	Block diagram of the proposed system	18
4.2	Frame with drawn in circle around the center of the image. Source: Bergerskogen’s report [2]	20
4.3	Polar conversion to make mapping easier	21
4.4	Example of distortion caused by interpolation during polar conversion	22
4.5	Overview of what happens with each frame through the system	24
5.1	Original and manually segmented reference frame	25
5.2	Weighting of grayscale image before Otsu	26
5.3	Difference between regular and weighted grayscale	27
5.4	Image with overlay	29
5.5	Difference between two frames with elevation angle lines from 10-90 degrees	30
5.6	Flaws with the two shapes of delta area.	31
5.7	Dilated difference to create delta area	32
5.8	Images showing the difficulties with finding three segments with SRM	34
5.9	Houghlines on videos with buildings and mountain	35
5.10	Original and segmented frames showing the difficulties with finding three segments using Otsu’s multilevel thresholding. 3 thresholds give 4 segments	36
5.11	Probability of blockages at different azimuth and elevation angles in different environments	37

5.12	Probability of LOS at different elevations with azimuth 0, 90, 180 and 270 degrees in different environments	39
6.1	Tracking of selected dots	44

List of Tables

3.1	verification in different types of environments	16
5.1	Comparison between weighted and non-weighted color components of the frames shown in Figure 3.2.	27
5.2	Threshold scale factor for best segmentation using Otsu's thresholding technique	28
5.3	Comparison between SRM and Otsu	29
5.4	Comparison between rectangular and dilated delta areas	32
5.5	Accuracy as frames are dropped	33

1. Introduction

This master thesis is written as a part of the Automation and Signal Processing program at the University of Stavanger (UiS). It was given as an assignment from the Norwegian Defence Research Establishment (FFI).

The thesis is structured as follows: Chapter 1 gives an introduction to the work and contains motivation and background for the thesis as well as a description of the datasets used. Chapter 2 is a theory chapter containing background theory on some of the image processing techniques used in the experimental part of the thesis. Chapter 3, *Proposed System*, gives a step-by-step description of the proposed system as well as explanations to each of the algorithms the system consists of. The experiments, results and reasoning behind the choices made in the proposed system are described in chapter 4. Chapters 5 and 6, *Discussion* and *Conclusion*, summarize the results and reflect on what can be done working further on the topic.

1.1 Motivation

Satellite communication (SATCOM) is a growing field of communications. Everything from TV transmissions to Global Navigation Satellite Systems (GNSS) such as GPS, GLONASS and GALILEO uses satellites to work. Some SATCOM systems utilize the Super High Frequency (SHF) i.e. 3GHz-30GHz. Communication in this frequency band often requires line of sight (LOS) to function properly; especially links with small antennas with limited link margins are vulnerable to attenuation due to vegetation and terrain. Lower frequencies have less vegetation attenuation, but will not have the same transmission capacity due to less available bandwidth according to the International Telecommunication Union's Radiocommunication sector (ITU-R)'s recommendations [3].

Knowledge of the environments around the antenna is important in order to best be able to utilize the communications efficiently and get a realistic view of expected performance. Information about the surrounding environments can be gathered via echolocation or manually writing a log, but the most commonly used method is to mount a digital camera beside the antenna to create a video of the antenna's coverage area. This video can then be processed manually or automatically to gather the information digitally. Along with the information on the surroundings, the information on signal quality from the antenna, and positioning and orientation data from a GPS and an inertial navigation unit should also be collected.

1.2 The objectives of this thesis

In 2014, the Norwegian Defence Research Establishment (FFI) did a SATCOM on the Move (SOTM) measurement campaign with the aim to explore the possibility for using this type of technology in Norway towards a geostationary satellite at X-band. X-band uses ranges of 7.9-8.4 GHz up-link and 7.25-7.75 GHz down-link. The main reason for the measurement campaign was to investigate the SOTM availability at X-band in Norway. The measurements were carried out by driving from Kjeller to Porsangmoen in the North, and to Bergen in the East, as shown in Figure 1.1. Because Norway is far north, the satellite will be closer to the horizon than in more southern parts of the world. The possibility is higher of having more obstacles in the transmission path here in forms of vegetation and buildings than if the measurements had been done closer to the subsatellite point.

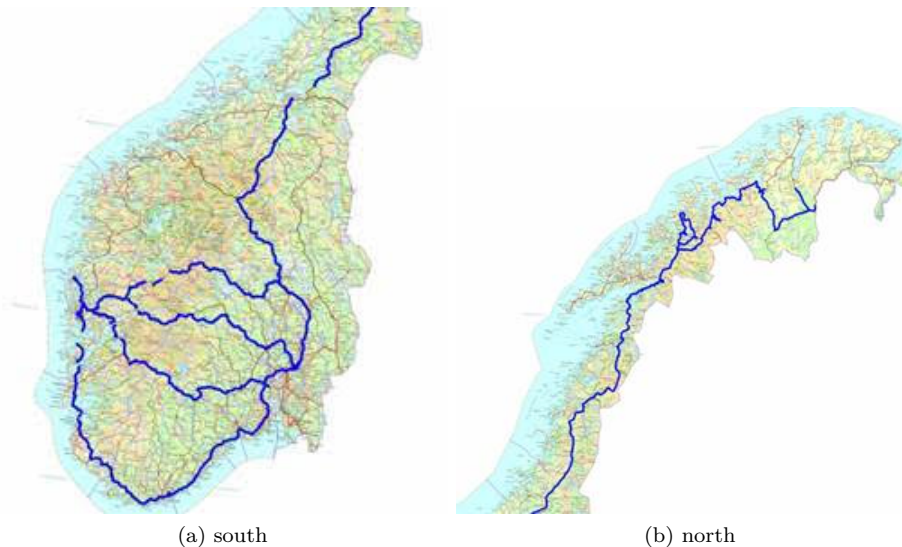


Figure 1.1: Driving route in measuring campaign

The signals from the satellite were recorded along with hemispherical videos of the sky. These videos is the main data set used in this thesis. The aim is to analyze the video and find methods to distinguish between sky, vegetation and solid objects like buildings and terrain in a three state model. The videos along with the information about the satellite position and signal properties can then be used to find what is causing attenuation or blockage in the satellite signal. This can later be used to set system requirements or create models or recommendations for communication with satellites at low elevation angles. It might be especially useful in communication with non-geostationary satellites as one can make a model of signal reliability at different elevation angles and set elevation angle requirements for such satellites for increased communications reliability.

1.2.1 Problem Formulation

The problem formulation from FFI, translated to English by the author of this thesis, reads as follows:

On the basis of the two earlier works [Bachelor thesis [4] and summer intern report [2], author remarks] an algorithm should be developed for automatically determining the cause of attenuation of the signal strength using the information available in video and log files (direction and elevation angles). The cause of attenuation should be classified as vegetation, buildings and terrain, or unknown/probable measurement error that requires closer inspection. This corresponds to the expected signal quality in the three states clear view, vegetation and fixed objects like buildings/terrain. For detected clear view, the signal is expected to be complete; for objects, the signal is expected to be noise only.

The main goal is to calculate the probability of the three states as a function of cardinal direction and elevation angle in various types of environment. The results are compared with earlier published measurement results. The main task will also address the methods used, and potential sources of deviation to the results should be quantified. The work can be performed outside of FFI's facilities and is not classified under the Security Act.

The text in its original language and form can be found in Appendix A.1.

In order to reach this goal, it has been broken down to several smaller objectives to be achieved through the thesis. These are as follows:

1. Verifying earlier works [2] and [4] for segmenting into two segments; LOS and NLOS.
2. Improving said earlier works. This will include transcending the algorithm from analyzing single frames to including the video aspect.
3. Creating a statistical model for probability of blockage at varying azimuth and elevation angles both in relation to the car and in relation to cardinal directions.
4. Separating the NLOS segment further into attenuating and blocking objects.
5. Updating the statistical model from objective 3 to include the separation of attenuating and blocking objects.

1.2.2 Limitations

The thesis will be limited to the development and improvement of software and algorithms, not the physical implementation or practical use of the product in the field. Therefore, subjects such as required hardware and financial constraints will not be evaluated. The scope of the thesis will also be limited to developing a system that can be used for weather conditions that are humanly interpretable; the system will in other words not be developed for weather conditions such as heavy fog with very short visibility and extreme low light conditions. The testing of the final product will be limited to the videos provided by FFI. No financial resources or high end data center capabilities for rapid analysis of large data sets were allocated to the work.

1.3 Background

1.3.1 Background Material

The assignment is based on two projects written for FFI in the past: A Bachelor thesis [4] and a report written after a summer internship [2]. The measuring campaign was carried out during the early summer of 2014, meaning that the Bachelor thesis used an older data set. The older set consisted of three videos recorded in and around FFI's offices in Lillestrøm, Norway. The summer intern had the possibility to use the new data set, but due to problems with the video format described in chapter 1.3.4, he used the old data set.

The aim of the previous works, was to differentiate between sky and non-sky parts of the video to be able to see if the satellite signals had any obstacles in its path. They did not focus on dividing the non-sky segment further in to vegetation and solid objects like buildings and terrain.

1.3.2 Previous Works

Finding satellites using photogrammetric techniques is quite widespread. Most of the papers on the subject researched for this thesis are based on two major measuring campaigns, MiLADY (Mobile satellite channel with Angle Diversity) that sought to use angle diversity in different environments to study fading from satellite systems and develop a fitting channel model [5] and CAPLOC (Combinaison de l'Analyse d'image et de la connaissance de la Propagation des signaux pour la LOCALisation) that sought to improve positioning in high density urban environments [6]. Both campaigns were conducted the same way as the one conducted by FFI in 2014 with an antenna and a hemispherical camera mounted on a vehicle together with a GPS receiver, power supply and laptop. A figure of the measuring system used in MiLADY is shown in Figure 1.2, but this setup is applicable to all the measuring campaigns mentioned in this thesis.

The majority of these earlier papers had the same aim: To improve SOTM and study the causes of attenuation, refraction and blockage of the signal in different environments. Bråten et.al did a statistical characterization of propagation environments in 2002 to find the probabilities of three different states (LOS, vegetation and solid objects) at different elevation angles as a function of azimuth and zenith angles [7]. They produced a two-dimensional state occurrence probability map for urban areas that could be a useful tool in trying to predict signal qualities at the different elevations. They did not produce one for other environments, but proposed that the development of such maps could be useful in further studies on the subject. Many earlier studies have studied the effect of different types of environments at different elevations, but few have factored in the driving direction of the measuring vehicle. This is studied in an article by Marie Rieche et.al. [8]. This article looked at the azimuthal direction of the satellite and how driving direction affected the signal quality. Generally in a right-hand driving country, the roadside objects are closer on the right side, this means that they will provide a bigger signal shadow than the left side. The report showed that the quality of the signal was best at azimuthal angles straight in front or straight behind the vehicle, with front-right and rear-right at

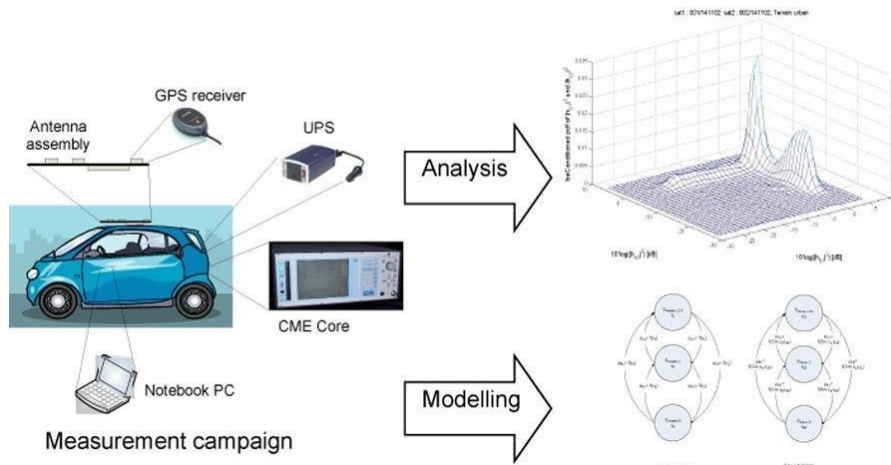


Figure 1.2: Setup of vehicle in measuring campaign. The image is taken from the MiLADY website[1]

second [8]. These results are not factored in to the proposed system of this thesis, but they are a point of improvement in future works described in chapter 6.4.

In using image/video processing to find satellites and the obstacles in the signal path, a few methods have been documented. Attia et.al. did a comparison between different supervised and unsupervised clustering techniques in 2011 [9]. The method they saw as most promising was a Geodesic Reconstruction by Dilation (GRD) followed by a Fisher's clustering algorithm. Statistical Region Merging (SRM) gave slightly better results, but with a computation time almost ten times slower than Fisher, Fisher was the most optimal for use in real-time processing. Because the proposed system of the thesis is meant as a post processing algorithm, the time constraint is not that important, so SRM has been tested as a viable option in this thesis.

Another way to go in classifying pixels into LOS and NLOS or further into vegetation and buildings/mountains, is a texture and color analysis. This is described in Meurie et.al.'s paper on the matter [10]. This paper seeks to combine texture and color information to find optimal weighting coefficients for segmenting using an adaptive segmentation method. A few different combinations showed promise in different kinds of images, but the one that worked best for the type of sky facing hemispheric images captured in the measuring campaigns was a combination of 50% color and 50% texture. These tests were primarily conducted in the RGB color space. With a fixed combination of color and texture, the YC_bC_r color space gave the best result and $L^*a^*b^*$ gave the best results when only using color, but with the adaptive segmentation method Meurie et.al. proposed, RGB worked the best. This has an added benefit of not having to convert and study other color spaces than the one the images/videos are captured in, thus saving time. Texture and color information is not used as much in the current master thesis, but they are seen as viable options in trying to segment the videos into vegetation and solid obstacles later.

1.3.3 Data sets

The main part of the data sets recorded in the X-band SOTM trials that are used in this thesis is the videos. The videos recorded have representation from many different types of weather conditions such as clear sky, partially cloudy, overcast and downpour. In addition to this, the videos contain different types of environments like urban towns, mountainous areas, forests and open fields. Examples from the videos can be seen in Figure 1.3

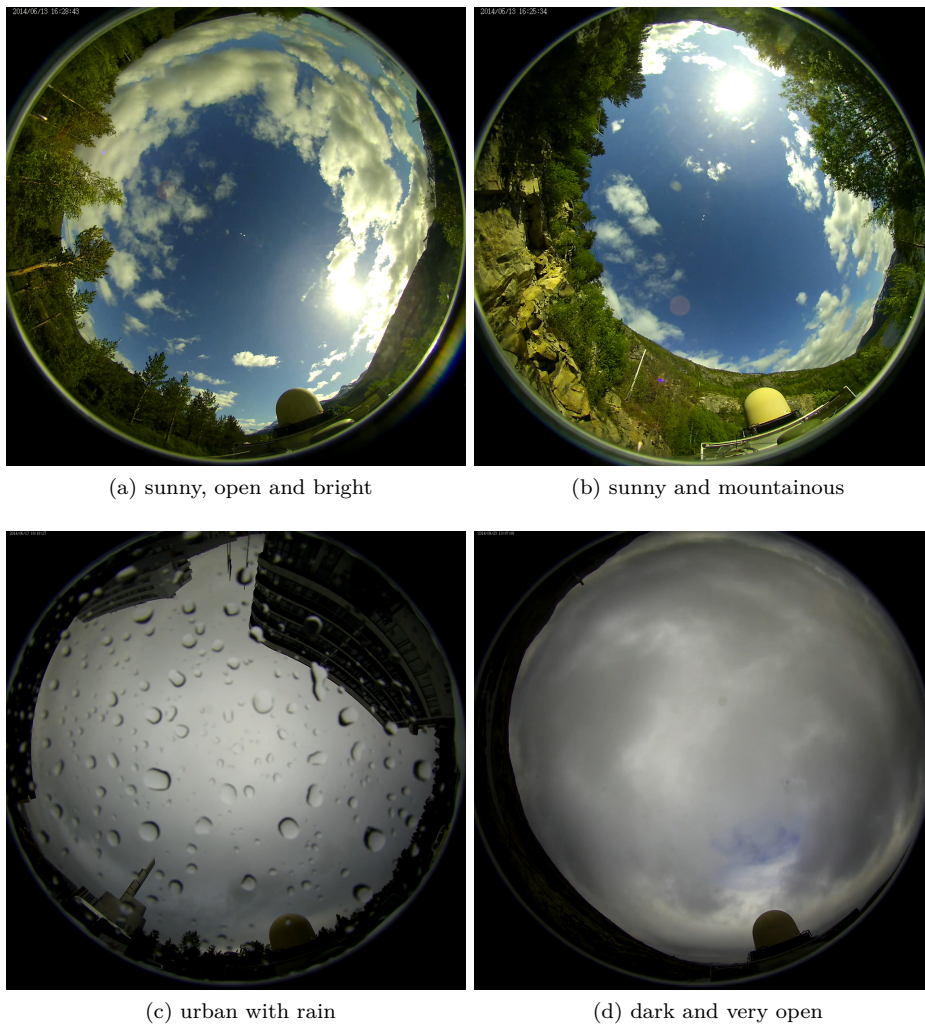


Figure 1.3: Example frames from different videos

The videos were recorded using a VIVOTEK FE8172V hemispherical camera [11]. The camera is made for use in the security sector for surveillance purposes, but it has several qualities that FFI assumed would make it suitable for the use described in this thesis. The most important feature they sought for was the hemispherical lens that made it possible to capture an almost 180 by

360 degrees view of the surroundings. A second important feature is the encapsulation of the camera. It has an IP66 encapsulation degree which means that it is waterproof and able to withstand almost all outdoor weather conditions. In addition to these features, the camera uses only one cable for recording, streaming, saving and power (PoE - Power over Ethernet) which made it easy to use when it is mounted on the roof of the vehicle. It also saved the video files as MPEG-4 which is a video format supported by MATLAB. However; transferring the videos to MATLAB has been a challenge. This is further explained in subsection 1.3.4. The camera was chosen by FFI because they saw it as the best option available on the market.

In addition to the videos the data set also contains a .txt file with information provided by the antenna and GPS in the vehicle. Some of this information includes position of the satellite, position and orientation of the antenna/vehicle, signal strength and signal to noise ratio (SNR). The data points in the file are time stamped and can thus be linked with the video to form a more complete image of what is happening as time progresses. This log file originally stems from an earlier test trip done before the measuring campaign started, and is thus only included to provide provisional data to the polar conversion and rotation blocks in the proposed system. Apart from this, the data is not included as a part of the analysis in the thesis.

1.3.4 Encoding of video format

The video provided by the camera is in the format MPEG-4 with an h.264 codec. This format was chosen because it is supported by MATLAB [12]; however when trying to import the video files into MATLAB the following error message was shown:

```
Error using VideoReader/init (line 619)
Could not read file due to an unexpected error. Reason: Error
Creating Source Reader Reason: The byte stream type of the given
URL is unsupported.
```

To get around this problem, the converter function of the multimedia player VLC (Video LAN Client) [13] is used to convert from MPEG-4 h.264 to MPEG-4 h.264; converting from the same format back to the same format. This enables MATLAB to read the file, but VLC has problems keeping up with the framerate when playing the file. Other downloaded sample videos with the same format and codec do not encounter the same error in MATLAB. No good explanation has been found for this, but it seems like it might have something to do with the way the codec is used. Further research into it was not prioritized. Making the videos readable was a minor, but integral part of making the rest of the study possible.

2. Theory

This chapter will cover some of the techniques and methods used in processing the videos. The theory chapter is deemed necessary to explain some of the methods used in this thesis one could not expect to be covered in a basic image processing course.

2.1 Image processing

2.1.1 Otsu's thresholding method

Otsu's thresholding method is a method for finding the threshold in a grayscale image proposed by Nobuyuki Otsu in an IEEE paper in 1979 [14]. It uses an unsupervised and non-parametric cluster based thresholding technique by going through every possible threshold and calculating the spread of the pixels within the foreground and background clusters respectively. The goal is to find the threshold that gives the lowest weighted variance within each cluster, σ_W^2 . The variance within the two clusters are weighted using $W_{b/f}$ to find the right proportion. Finding the Within Cluster Variance, σ_W^2 , is quite computationally intensive, so an easier approach is to find the threshold that maximizes the Between Cluster Variance, σ_B^2 . It gives the same resulting optimal threshold, but is a lot simpler to implement and a lot less time consuming. This is the approach most implementations uses, including MATLAB.

$$\mu = W_b\mu_b + W_f\mu_f \quad (2.1)$$

$$\textit{Within Cluster Variance} \quad \sigma_W^2 = W_b\sigma_b^2 + W_f\sigma_f^2 \quad (2.2)$$

$$\begin{aligned} \textit{Between Cluster Variance} \quad \sigma_B^2 &= \sigma^2 - \sigma_W^2 \\ &= W_b(\mu_b - \mu)^2 + W_f(\mu_f - \mu)^2 \\ &= W_bW_f(\mu_b - \mu_f)^2 \end{aligned} \quad (2.3)$$

Otsu's method has a couple of advantages over other thresholding algorithms; first and foremost the fact that it only uses simple statistical measures such as mean and variance making the computational complexity a lot lower and thus faster.

2.1.2 Optical Character Recognition

Optical Character Recognition (OCR) is an algorithm that finds letters or numbers in an image [15]. In this thesis it is used to find the date from the time stamp in the upper left corner of the video. This is done because the metadata containing record date amongst other things is not included in the video files. The algorithm takes the region of interest from an image as input and outputs where in the image the text is placed and what it says. First the OCR algorithm usually goes through a preprocessing phase where the image is sharpened, de-skewed, non-character objects like boxes and lines are removed, and the image is often converted into black and white. There are two main ways of finding characters: Matrix matching and Feature extraction. An example of the two can be seen in Figure 2.1.

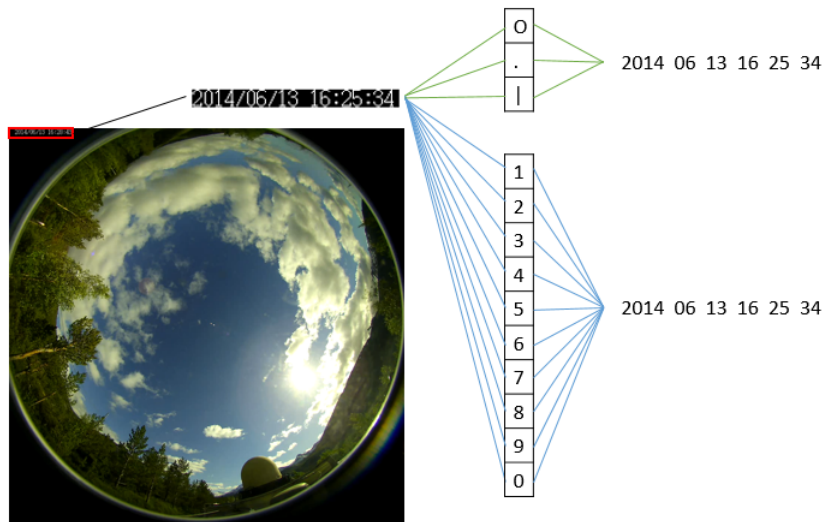


Figure 2.1: OCR with Matrix matching (blue) and Feature extraction (green)

Matrix Matching

Matrix matching compares each character to a database with reference characters pixel by pixel. The results are put in a ranked list of which character match the best. This method is dependent upon the character being isolated from the rest of the word correctly as well as each character box being the same size with the whole character fitting within one box.

It is an easy method to implement, but does not work as well with certain fonts where each character has a different height or width, or fonts that have parts of characters going over/under the previous/next. For example some fonts place slim characters such as 'i' and 'l' closer together to save space, or have the line in 'j' and 'y' go under the previous character. This saves space, but makes it more difficult to recognize using matrix matching because each box is a different size and some characters extend beyond their own box. [16]

Feature Extraction

Feature extraction looks for features such as lines, loops and dots. It also finds directions of different features and intersections between them. The features are then compared to a database with a vector representations of each reference character. Vector representations themselves are a composition of standard mathematically describable shapes, so comparing the extracted features with the vectors makes for a more robust OCR algorithm. [16]

2.1.3 Hough transform

The Hough transform is a tool used for finding straight lines or curves in an image. [17] It works by doing an edge detection and thereafter tries to find points on the edges that line up to form a straight line. The lines are then saved to a matrix with details on orientation (θ), length, and distance from the center of the image (ρ). The matrix is combed through to eliminate any lines that consist of too few points to be actual lines. The lines that are left are displayed as on top of the image as shown in Figure 2.2.

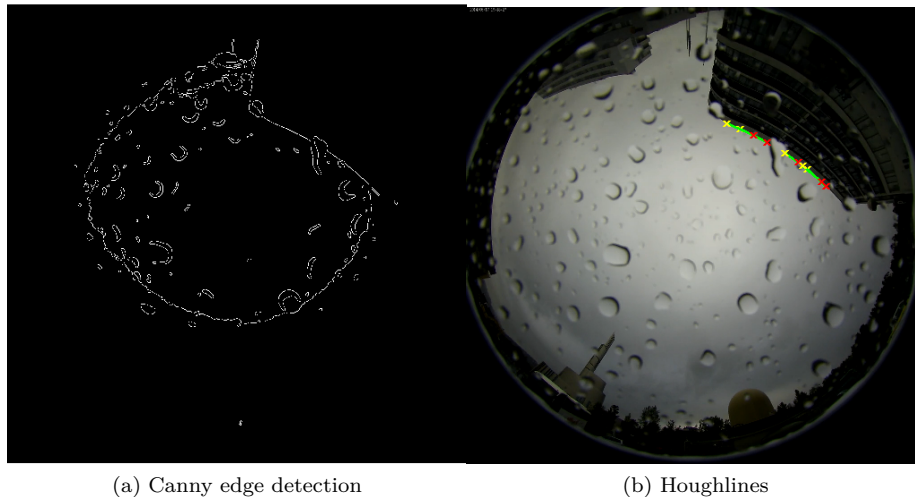


Figure 2.2: Hough transform applied to video frame with buildings

By setting a similarity requirement of how similar two pixels need to be to be regarded as part of the same line, and how many pixels are needed in order to form a line, the sensitivity of the transform can be controlled. If the sensitivity is set too low, only the strongest line, if any at all, is displayed. If the sensitivity is set too high, the image will become noisy with all the random lines that statistically will occur in any image.

2.1.4 Statistical Region Merging

Statistical Region Merging (SRM) is an algorithm proposed by Richard Nock and Frank Nielsen in the “*IEEE transactions on pattern analysis and machine vision*” to try to better the statistical errors most earlier segmentation algorithms make [18]. Most of these errors are with border pixels being wrongly assigned; overmerging where one observed segment contains more than one true segment, and undermerging where one true segment gets divided into more than one observed segments. There are also hybrid cases where an observed segment partially spans over many true segments, but does not include all of them completely.

The algorithm is easy to use and can be adapted to most color spaces, but throughout the thesis, the color space RGB will be used.

The algorithm is built on the homogeneity property that in a perfectly segmented image, the pixels within one segment have the same statistical expectation for any given color channel. The adjacent segments have a different expectation in at least one color channel [18]. When a pixel is sampled, the color channels are replaced with a set of Q independent random variables. These random variables are positive and bounded by g/Q such that they belong within the interval $\{1, 2, \dots, g\}$. The value g is the maximum value of each color channel. With standard RGB, $g = 256$, but with other color spaces it may differ. The introduction of Q is motivated by the ability to control the statistical complexity of the algorithm, the coarseness of the segmentation and the generality of the model. A figure showing the resulting image from an SRM segmentation is shown in Figure 2.3.

SRM consists only of two parts: A merging predicate and an order to test the region mergings in. The calculations of why these are as they are can be found in detail in Nock and Nielsen’s article [18]. This article uses the word “*region*” instead of “*segment*” or “*cluster*”. For color images with a region R with average values \bar{R}_a in each color channel a , the merging predicate is as follows:

$$\mathcal{P}(R, R') = \begin{cases} \text{true}, & \text{if } |\bar{R}'_a - \bar{R}_a| \leq \sqrt{b^2(R) + b^2(R')} \\ \text{false}, & \text{otherwise} \end{cases} \quad (2.4)$$

$$b(R) = g \sqrt{\frac{\ln |\frac{\mathcal{R}_R}{\delta}|}{2Q|R|}} \quad (2.5)$$

The order of testing the mergings follows an invariant \mathcal{A} which only states that when any test between two regions occur all the tests inside each of the regions have previously occurred.

With these two in place, the algorithm is almost good to go. First the image gets divided into pixel pairs following the 4-connectivity (one pixel gets paired in order with the pixel above, below, to the right and to the left). This makes the number of total pairs $N < 2|I|$ sorted in a set S_I in increasing order of $f(p, p')$ which is a real-valued function. The order is then traversed once and the merging predicate $\mathcal{P}(R, R')$ is tested on each pair as long as $R(p) \neq R(p')$. The two pairs are merged if the predicate returns true. The goal is to find the best function $f(p, p')$ to satisfy the invariant \mathcal{A} and make the sorting optimal for the merging to happen. The biggest difference here between SRM and other

clustering algorithms is that the order of the pairs and regions will remain as it is. It will not be rearranged and sorted for each new iteration. This makes the SRM run faster, and with a good choice of function, it will also run smoother and with less error. In their article, Nock and Nielsen state that there are many ways of choosing a good function, but that the easiest and most basic one is to find the difference between the two pixels:

$$f_a(p, p') = |P_a - p_a| \quad (2.6)$$

Other choices are modifications of the Sobel and Prewitt convolution kernels normally used for edge detection. Because SRM is based on 4-connectivity, only the middle row and column of the kernels are of interest when these are used.¹

The code used in implementing this system and translating it from the report by Nock and Nielsen and into a usable MATLAB code was made by Sylvain Boltz [19].

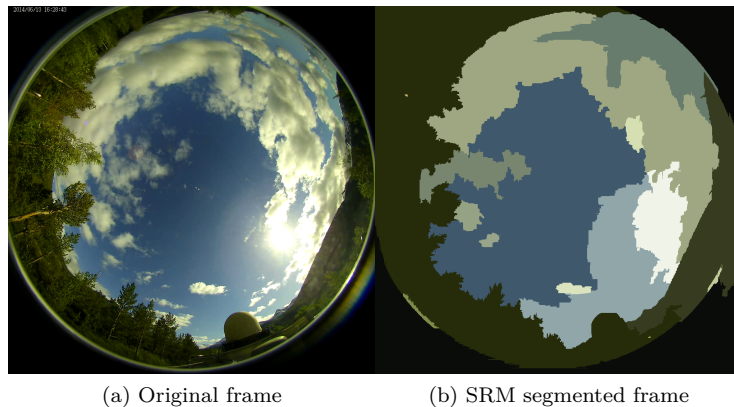


Figure 2.3: Statistical Region Merging on sunny frame

¹Fragments of this subsection have been extracted from previous works by the author of this thesis

3. Verification of Baseline Work

This thesis is meant to be a continuation of the report written by Pål Bergerskogen [2] which in turn is a continuation of the Bachelor thesis the author of this thesis wrote together with Ruud and Skoglund [4]. A part of the work in this thesis has been to verify those earlier works.

3.1 Bergerskogen's work

Bergerskogen did a verification and improvement of the method developed in the Bachelor thesis. A visual representation of his method is shown in the block diagram in Figure 3.1. His main additions were the Optimiser and Polar correction blocks. In adding these two, he also made the Initiation block a lot bigger because it now has to initiate all the polar conversion variables. The blocks are explained in this section, but as many of them are reused with minor changes in the new proposed system, a deeper explanation of each block can be found in the next chapter.

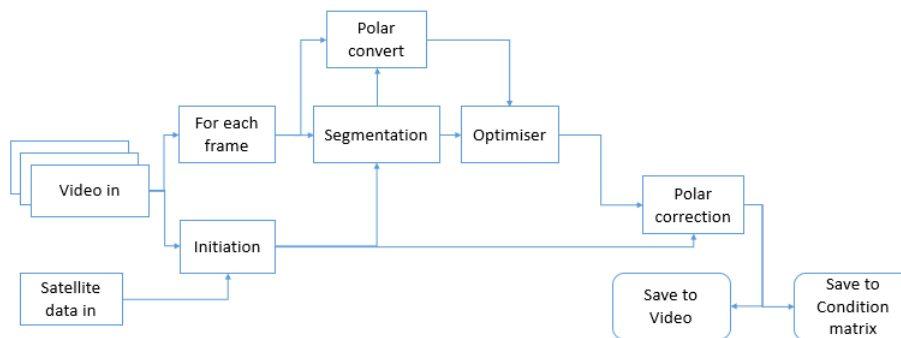


Figure 3.1: Block diagram of Bergerskogen's algorithm

Initiation

The initiation block takes user input, the video and satellite data as input and outputs polar conversion matrices. The user input consists of start date and time, and choices of precision and range in the polar conversion.

SegmentationMethodHIG

The SegmentationMethodHIG block is a narrowed down version of the system proposed in [4]. It uses Geodesic Reconstruction by Dilation (GRD) to filter the frame and remove noise, and the `Vision.Autothresholder` tool in MATLAB to finally segment the frame. `Vision.Autothresholder` uses Otsu's method for finding the best threshold. A threshold scale factor is multiplied with the threshold value computed by Otsu's method. The resulting value will be the new threshold value [20]. The default threshold scale factor is 1. This value can be changed, and through testing it was shown that a scale factor of 1.5 gave the best results on the specific video Bergerskogen used. However with just a small increase in scale factor beyond this, the accuracy went down drastically. To ensure that a proper accuracy was produced, the threshold scale factor is set to 1.3 in SegmentationMethodHIG.

Optimiser

The segment results from SegmentationmethodHIG have some holes of sky in the vegetation and vice versa because sparse vegetation is not dark or thick enough to be considered as NLOS by the Otsu's thresholding. Because we want even sparse vegetation to be classified as NLOS, the Optimiser block will reconstruct and fill in smaller white areas and thicken weak vegetation through active contours to make the two segments more definite and separate from one another.

Polar conversion

The videos were captured with a hemispherical lens. As the end product should work by mapping the satellite position in to the video as a function of azimuth and elevation, the frames are easier to work with if they are converted to polar coordinates. This means "folding them out" to wide angle horizontal images. An example of this can be seen in Figure 4.3.

The front of the vehicle is always at the top of the frame. To get a correct view of the surroundings in relation to cardinal directions, the polar video therefore needs to be shifted from showing the front of the vehicle as north, to showing the actual cardinal direction. How much the video needs to be shifted is calculated using positioning data from the log file.

End blocks

The results from the segmentation is saved to a video for easier to get a visual indication of how well the segmentation works. It is also saved to a condition matrix saying if the satellite is within LOS at any given time. This gives the user the opportunity to go back in time and check the reason for a bad or lost signal.

3.1.1 Limitations

Due to the video format problems described in subsection 1.3.4, Bergerskogen only developed an algorithm that worked on one video, i.e. only one type of

weather, environment, lighting etc. In addition to this, the author seemed to only train the algorithm on three frames throughout the video. Therefore, the first step in the verification process is to make sure the algorithm works on other frames within the same video. The next step is to check if it works with other similar videos, and lastly if it works with videos of different types of weather, lighting and environment.

3.2 Verification

The segmentation method found by Bergerskogen to be the best was what he called `Optimiser_05_1_4_10`. This method uses the `SegmentationMethodHIG` block along with the optimiser described in section 3.1 above. The numbers 05, 1, 4 and 10 denote the angle precision, structure element size, number of reconstruction iterations, and number of active contour iterations respectively. These are the inputs to the optimiser block. The verification process was started by applying this method to the different types of weather conditions, environments and seasons present in the new data material. Examples of the different types of environments can be seen in Figure 3.2.

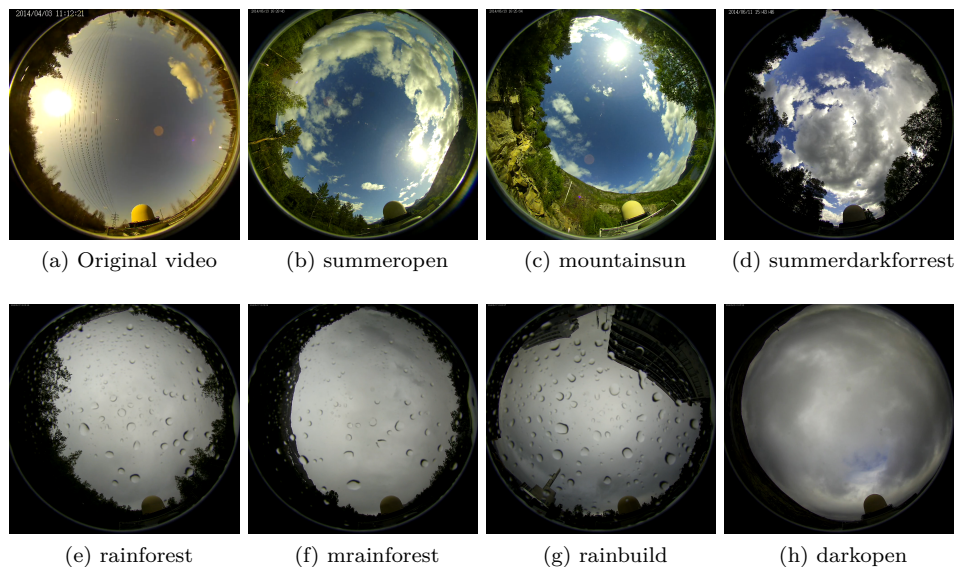


Figure 3.2: Examples of different kinds of environment

Bergerskogen developed a MATLAB program for performance checks which takes the video, a manually segmented frame and a preferred segmentation method as input, and outputs a .txt file with accuracy, percentage of correctly classified pixels, wrongly classified pixels, and elevation angle intervals at which the wrongly classified pixels are placed. The program was used by Bergerskogen to analyze the performance of different segmentation methods in the same video, but can be modified to be used for comparing the performance of the same method on different videos.

The area that is most difficult to segment is the transition between sky and vegetation, especially in the videos captured earlier in the year when the leaves have not grown out to make the trees fully opaque. Because of this the low elevation angle accuracy is included in the verification tables to see how well the algorithm performs in the transition zone. The results from the test can be seen in Table 3.1

Data Set	Environment description	Overall accuracy	Low elevation angle accuracy
Original	sunny, open, colorful	98.10	92.36
summeropen	sunny, open, colorful	97.89	92.45
mountainsun	sunny, mountain, colorful	96.96	92.91
summerdarkforest	dark, forested, blue sky	97.41	91.39
rainforest	rainy, forested, gray	95.94	85.00
mrainforest	rainy, forested, mountains	96.18	76.21
rainbuild	rainy, urban	95.36	72.92
darkopen	gray, open plain	96.24	71.15

Table 3.1: verification in different types of environments

Bergerskogen’s algorithm does well with other types of environments. Neither of these results are really bad over all. It is however evident that the algorithm was trained for and meant to be used on videos with green trees, blue skies and a visible sun. It is especially evident in the low elevation angle accuracies. As soon as there are raindrops on the lens or the color saturation drops, the accuracy of segmentation in the transition zone goes down.

The accuracy measure used is the basic number of correctly classified pixels divided by total number of pixels. It was done on the frames that have manually segmented reference frames. This is a good indication of how well the algorithm works, but to be sure, a visual verification was done as well. The segmented video was put on top of the original video to create an overlay. This video was then watched to manually assess how well the algorithm works outside the 4 reference frames. Some selected stills from the overlay video are shown in Figure 3.3.

The videos that were shown to have a good accuracy in Table 3.1 did have a good accuracy in the visual verification sampled in Figure 3.3 as well. Most of the wrongly classified areas were underneath power lines, around lampposts or above tall structures as shown in the top and middle image of the figure. Over all the segmentation algorithm worked well on the bright and sunny videos, as one would expect from an algorithm that is trained on this type of video.

The bottom image in the figure is from one of the rainy and gray videos. All of these were wrongly segmented because the sky is dark close to the ground, so it will end up under the threshold being segmented as NLOS.

An easy fix to this would be to shift the threshold scale factor of the Otsu’s seg-

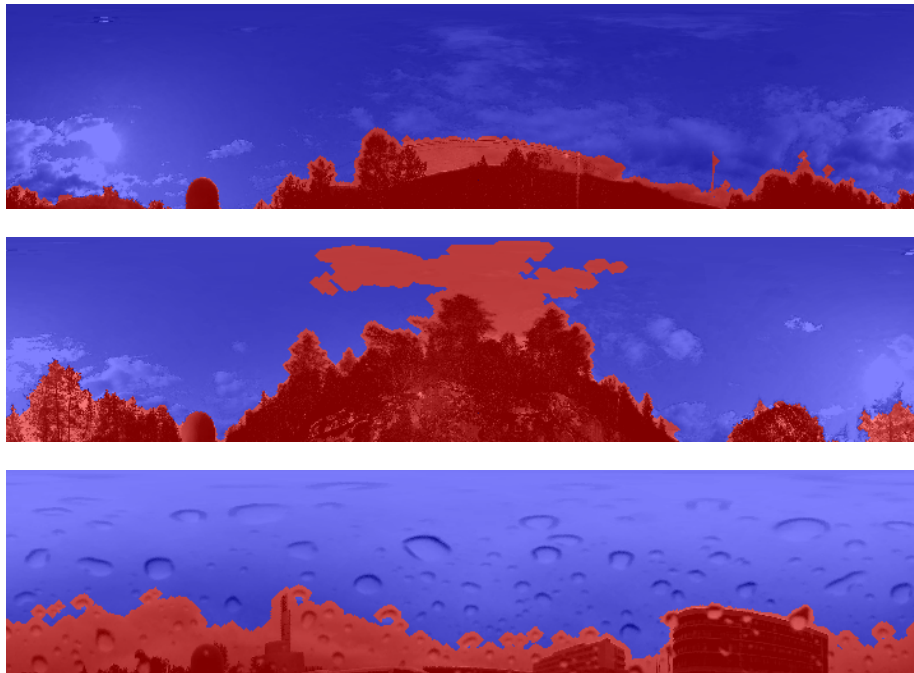


Figure 3.3: Selected stills from the visual verification videos. Top: summeropen, Middle: mountainsun, Bottom: rainbuild

mentation. That would move the segmentation boundary closer to the actual boundary between sky and buildings, but it would most likely not get around the raindrops that are visibly distorting the boundary between segments in the bottom image.

Knowing where Bergerskogen's solution is flawed or limited will help in improving the algorithm further in this thesis. A possible way to do this based on the aforementioned visual verification is to look at different thresholds and make them dependent of the weather type in the video. This is further explained and tested with good results in section 5.3 of the Experiments and Results chapter.

Other elements of Bergerskogen's solution that need verification are what he wrote he was unsure if he understood or implemented correctly. One of these things is the the correct calculation of satellite angle in relation to the GPS position and angle given by the log file. He found a "satellite calculator" online, but did not verify the results it gave or reference where he got the calculator from. The azimuth and elevation angle he calculated for an antenna in Kjeller, Norway to a geostationary satellite at 6 degrees West was 184.9Az and 21.6El. This seems reasonable when considering the two points in relation to one another. And when factoring in several other online and offline satellite calculators [21][22][23] saying that the true azimuth and elevation are 185.2Az and 21.8El, Bergerskogen's calculation method for satellite position in relation to the recorded GPS positions is within reasonable accuracy.

4. Proposed system

The system proposed is a series of different modules. Each of these are shown in Figure 4.1. An overview of what happens with each frame throughout the different modules is shown in Figure 4.5. It is based on the system proposed in [2] and there are some similarities, but this proposed system has had some bigger changes in structure, efficiency and accuracy. Each of the blocks are described in this chapter, and the testing and choices behind the blocks are described in chapter 5. The full MATLAB code can be found in Appendix A.4.

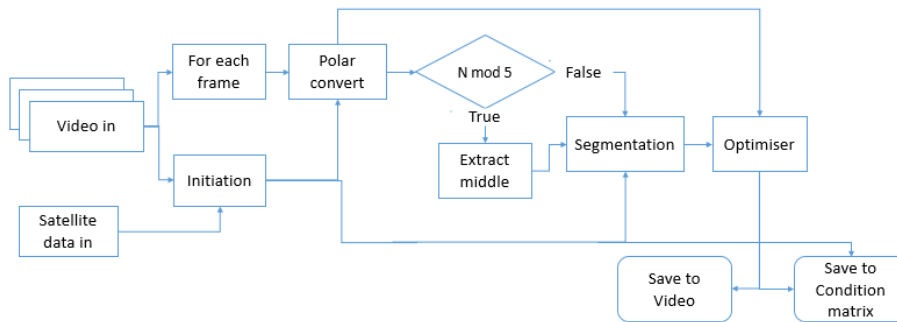


Figure 4.1: Block diagram of the proposed system

4.1 Initiation block

The initiation block reads the frames and determines the type of environment. It also initiates all the variables needed to transform from the circular image to the rectangular one. This is further explained in block 4.2.1.

4.1.1 Input

The input needs to be of the right format. The satellite data is provided from a log file made by the antenna. The log file used in this section is not the one belonging to the videos, but it should be of the same format, and is only included to provide a provisional basis for the polar conversion and rotation. This file shows information on the signal strength and SNR from the satellite, and GPS and inertial navigation unit data telling the position and orientation of the vehicle. The file's recorded data spans over several hours, so later in the initiation the relevant data is extracted for use in the rest of the system.

As explained in section 1.3.4, the raw video file from the camera is unreadable in MATLAB. Therefore the videos need to go through a conversion to be readable. This is explained further in section 5.1. After the necessary conversion, the video can be read in to MATLAB by the user writing start time and date as the videos are named on the format *sotm_YYYYMMDD-HHMMSS.mp4*.

4.1.2 Finding environment type

Before the real segmentation and classification of the main parts of the video begins, a short snippet of code determines what kind of weather, environment and time of year the video is set in. This is because different conditions, especially in weather, will change the looks and color properties of the frame or video. In summer the trees will be green, the sky can be blue, the chance of sun glare is high and shadows become very sharp. In winter everything looks gray, the trees have no leaves, and the transition from white snow to white or gray sky is hard to spot. To make the classification algorithm work as good as possible in every condition, it must be prepared for what to look for; which properties should be weighted the most and where the different thresholds should be set.

To find the type of environment the video is captured in, an OCR algorithm was tested to read the timestamp in the top left corner of the video. This however was quickly deemed obsolete as the videos are named with a timestamp, and the user will already have had to input the timestamp to be able to read the video in the first place. Even so, it was useful on the training sets where the videos were named after weather type rather than recording date as explained in the next chapter.

4.1.3 Weighting of color components

Otsu's method takes a grayscale image as input and outputs a black and white image. Normally the grayscale image is just a regular intensity image where each color (R, G and B) is weighted to make the image more pleasing to the human eye, but with multiplying each color channel with a different scaling factor (shown in Figure 5.2), the resulting grayscale image gives a better starting point for Otsu's method making the end result better. The choice of scaling factors is based on testing done in section 5.2.

4.1.4 Polar conversion variables

This part of the initiation block is almost the same as Bergerskogen's initiation block. The position of the car in latitude and longitude along with roll, pitch and heading are read from the antenna log file. This log file has recorded data with an inconsistent sample frequency for hours on end, so part of the initiation will be to find the start time and duration of the video to extract the relevant information from the log file. This data is then interpolated to fit the framerate of the video.

The heading and position of the vehicle will tell how it is situated in relation to the cardinal directions. Because the videos always record the front of

the vehicle as 0 degrees azimuth i.e. "north", the results of the analysis will be wrong for all driving directions other than the vehicle driving north. To correct this, the initiation block calculates the shift needed in the polar converted frames as a function of time to factor in driving direction in the calculation of true azimuth. The user can decide the precision and lower limits of the polar conversion variables in the beginning of the initiation block. The precision is set to 0.5 degrees to save time and space, and the lower limit for the elevation is set to 10 degrees. This choice was made by Bergerskogen and his project manager because the surroundings at this low elevation are mostly irrelevant to the study [2].

A small correction that also happens in this block is the correction of the middle of the lens. Figure 4.2 shows a frame from Bergerskogen's video with a circle drawn in around the center of the frame. It does not line up with the center of the circle from the camera meaning that the center of the frame is not exactly at 90 degrees elevation. It should be possible to find the center of the lens, i.e. the true zenith, with image processing techniques like `houghcircles` or the `regionprops` function in MATLAB, but Bergerskogen made a provisional solution where he counted the pixels and found the offset to be 19.5 pixels by 11.5 pixels, and using this has worked well [2]. This offset is also factored in to all calculations of polar conversion.



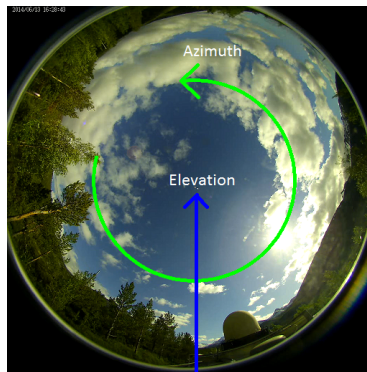
Figure 4.2: Frame with drawn in circle around the center of the image. Source: Bergerskogen's report [2]

4.2 Main blocks

4.2.1 Polar conversion

The videos are captured by a hemispherical lens. This gives them a visual layout that is more difficult to work with. The end product is supposed to work by mapping a satellite position to the frame as a function of azimuth and elevation. This is not easily done in the hemispherical image where 90 degrees is in the middle of the image while the edges of the circle range from 0 to 360 degrees. To make the frame easier to map, it is "folded out" and converted into polar coordinates. This can be seen in Figure 4.3. The new frame will have azimuth along the θ -axis and elevation along the ρ -axis.

θ goes from 0 to 360 degrees with 0 being upwards in the frame. ρ goes from 10 to 90 degrees with 10 being almost at the ground. As described in the last section, the lower limit of ρ can be set by the user if they want to change how low towards the horizon the analysis of the video should go. To make these correct with cardinal directions, an offset is added to each of them based on the information about position and heading extracted from the log file in the initiation block.



(a) Hemispheric frame



(b) Polar frame

Figure 4.3: Polar conversion to make mapping easier

This way of displaying the frame also makes it easier to separate parts of the video that don't change as much. Both the top sky and the bottom ground stay within the same segment through the whole video. Knowing this will help bring down the computation time as these parts can be excluded from the segmentation algorithm because they don't change. A comparison of two techniques utilizing this option is done in section 5.5 of the Experiments chapter.

A drawback to this method is the fact that the top part (middle of the sky) has to be interpolated. As there is usually only sky there, an interpolation will not usually have that big consequences for the accuracy of the segmentation. If an object like a powerline or a raindrop is observed in the middle of the original frame, it will appear long and distorted in the polar converted frame which may have an impact on the performance of the segmenting algorithm as a whole. However the likelihood of this happening and having a major impact is small. All in all the benefits of polar converting outweighs the consequences.



Figure 4.4: Example of distortion caused by interpolation during polar conversion

4.2.2 Segmentation

Otsu’s method was found to be the most promising segmentation algorithm in [4] and [2]. It is described in section 2.1.1 of the Theory chapter. What happens in this block is that the weighted and polar converted graylevel frame is put through a GRD filter to filter out noise and make the frame easier to segment. It is then put through the `Vision.Autothresolder` tool in MATLAB which uses Otsu’s method to find the optimal threshold. As explained in section 3.1, `Vision.Autothresolder` also has a possibility to add a threshold scaling factor (1 by default). This scaling factor is then multiplied with the optimal Otsu’s threshold to produce the final threshold the frame is segmented with. The choice of threshold scaling factor for different types of weather is described in section 5.3, but it generally lies around 0.8. The actual segmentation is done with a `step` function that takes the autothreshold and frame as input and outputs the segmented frame.

The proposed system also makes use of the Optimiser block constructed by Bergerskogen and explained in section 3.1, which closes holes and makes the two segments more definite and separate from one another.

4.2.3 Optimization

The system proposed by Bergskogen [2] segmented each frame fully and put them back together as a video afterwards. This works well, but with a framerate of 3 fps and the car the camera is mounted on driving at approximately 50km/h, the video won’t change that much between neighboring frames. Therefore there is no use in segmenting the whole frame for each and every frame. In just segmenting the parts of the video that changes, the algorithm as a whole will become more effective and a lot less time consuming. The time overall has been reduced by 36% compared to Bergerskogen’s system which is a big improvement. This is described further in section 5.5.

The area within which the video that changes is found by segmenting the first two frames and subtracting them from each other. The resulting area is the difference between the two frames. With high probability the next frame will differ from the two prior ones in approximately the same area, so this is in theory the only part of the next frame that needs to be segmented. To find this area the difference between the two first frames is morphologically dilated to create a wider band to be sure to catch the change in the next frame. The

next 4 frames are only segmented within this band. The segmented areas are put on top of the first frame to make a complete frame to save to the video and condition matrix. The process is repeated every 5 frames with the 5th frame being segmented fully and the rest just in the resulting band. The choice in band size and interval between frames being segmented fully is based on experiments described in section 5.5 of the Experiments chapter. A full view of what happens with each frame as the system progresses is shown in Figure 4.5.

4.3 End blocks

4.3.1 Save to Video

As each frame get processed and segmented, they are added to an AVI (Audio Video Interleave) video file. This file is opened for writing in the initiation block, and frames are added as they get segmented. The file is then closed after the whole video has been processed.

An option in this block is to save the new segmented frames as an overlay on top of the original video. This is done by fusing the frames one by one using the MATLAB function `imfuse` and saving them to the video file. A toggle switch to do this is in the user input part of the initiation block. The overlaid video is a useful tool in manually getting a sense of the performance of the algorithm and which parts of the video the different methods might succeed or fail in classifying correctly.

4.3.2 Save to Condition Matrix

After the frames have been segmented, they are saved to a matrix called *frame-Cell*. From this matrix each frame is compared with the position of the satellite from the position matrix in the initiation block to see if it falls in the LOS or NLOS area. The results are saved as ones and zeros in a condition matrix along with the respective timestamps. The reason for this is to give the user the option of inquiring about the state at different times where they might have seen the signal quality dropping. The program has a section in the end asking the user for a time, and when received going through the condition matrix to return the state at that specific time. In the future, this could be made more user friendly and GUI based rather than command line based.

The condition matrix is also used in generating statistics as is proposed in the problem formulation. It will obtain information on every frame throughout the video as well as whether the satellite was within a LOS or NLOS segment at any given time. This can be used in finding the proportion of LOS to NLOS in a certain environment, or the probability of blockage at different azimuth and elevation angles, and in different environments as described in section 5.7. Because the log file containing the positioning data is from another set of videos, the statistical data found through the condition matrix can not be used for the proposed statistical purposes in this thesis, but as soon as the correct log file is obtained, the condition matrix will be of big help.

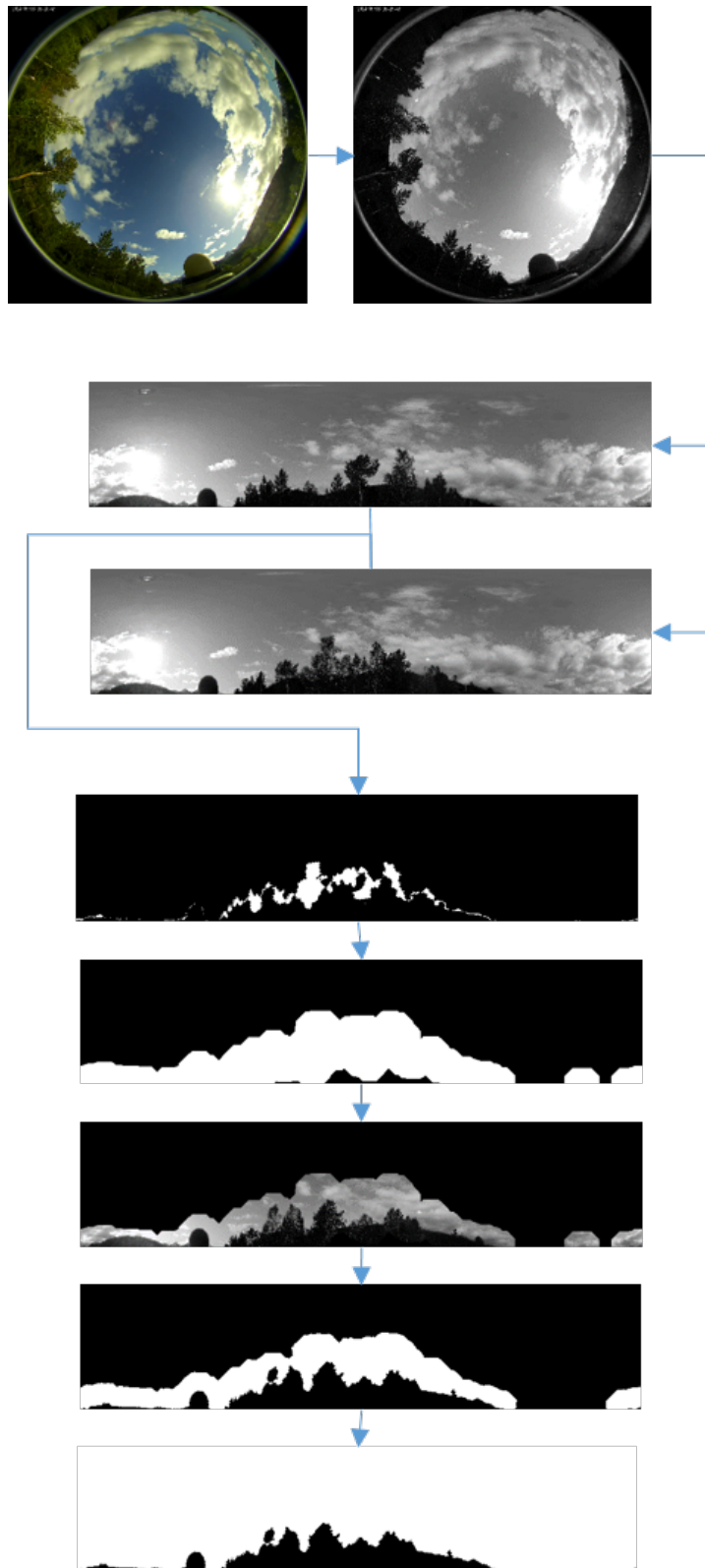


Figure 4.5: Overview of what happens with each frame through the system

5. Experiment and Results

5.1 Preparatory work

Before the video and satellite data are loaded in to MATLAB, a few things have to be done to prepare the data.

Converting video format

As explained in subsection 1.3.4, the video format provided by the camera has been unreadable to MATLAB. To convert these to a readable format, the conversion tool in VLC is used. The original videos are also quite long and have been shortened using VLC's record option. All the new videos used for testing are about a minute long making the file size a lot more manageable. The new videos have also been given names reflecting what kind of weather they contain. This will be changed back to the original *sotm.YYYYMMDD-HHMMSS.mp4* as soon as a reliable weather recognition algorithm is found.

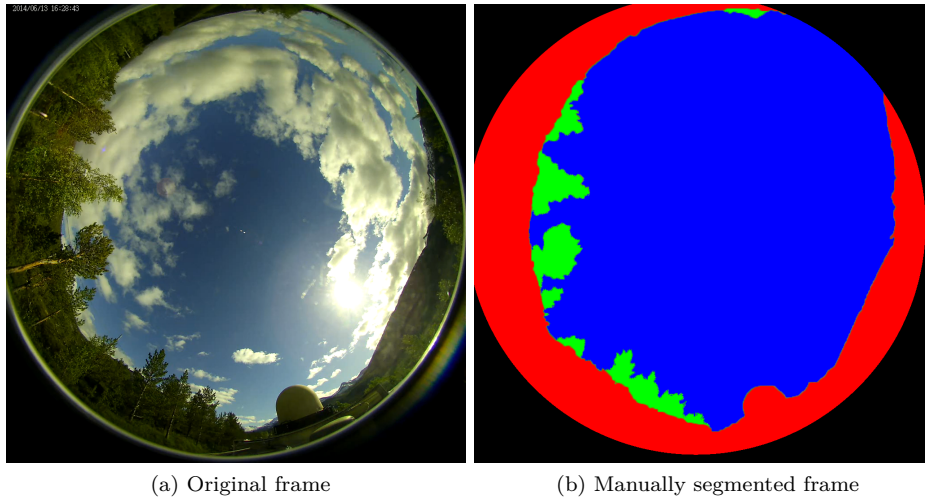


Figure 5.1: Original and manually segmented reference frame

Reference images

To test the performance of each method, selected frames from each video were manually segmented to create a set of reference images to compare the automatically segmented images with. The manual segmentation was done in Photoshop and an example is shown in Figure 5.1. This was done for 4 frames from each video shown in Figure 3.2 except the first two. The first video in Figure 3.2 (*Original*) is the video used by Bergerskogen where he made 3 reference frames. From the second video (*Summeropen*) 3 frames throughout the video in addition to a series of 10 frames were manually segmented for use in checking the performance of the optimization which required a reference video sequence rather than stills spaced throughout the video. The manual segmentation of frames takes a lot of time, which is why there were not made more reference frames for each video. The repercussions of this are further discussed in section 6.1.1.

5.2 Weighting of color components

As mentioned in section 4.1.3, multiplying the different color channels with scaling factors before using Otsu’s method gives a better starting point compared to using the `rgb2gray` function in MATLAB or the intensity image. The multiplication is done as shown in Figure 5.2. It is possible to use other color spaces than RGB too, but due to time constraints, this was not tested in this thesis.

The result for the different videos from which the frames in Figure 3.2 are

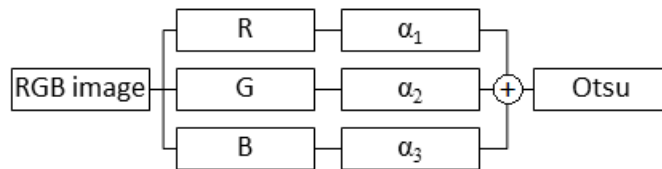


Figure 5.2: Weighting of grayscale image before Otsu

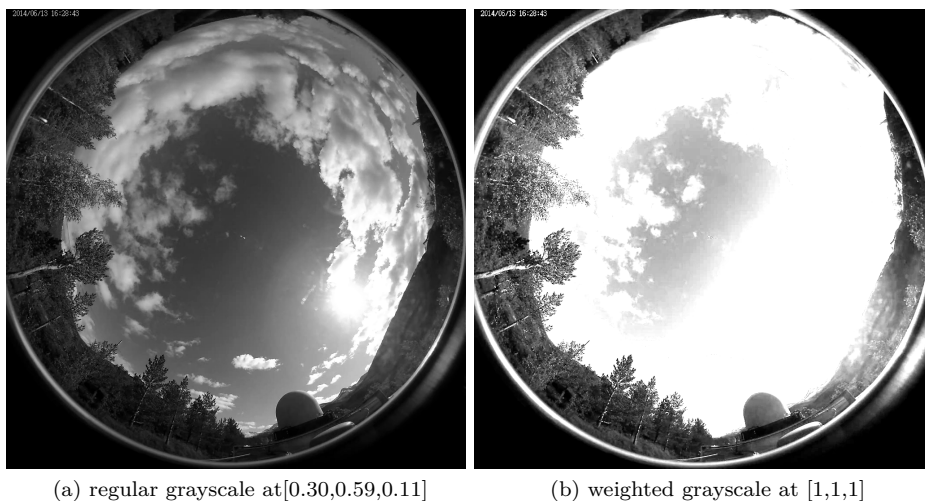
gathered, are shown in table 5.1

Weighting the different color channels had some effect on the outcome of the thresholding algorithm. In some of the videos it had a big effect while the gain was smaller in other videos. For separating sky and non-sky it is apparent that the blue channel of the original frame is where most of the information lies as it is weighted at 1 or close to 1 in every frame while the red and green are weighted lower. This is especially visible in the colorful videos with the sun shining, the sky mostly blue and trees mostly green.

Some of the frames were segmented better with weighting even though the weighting variables were 1,1,1 i.e. instinctively you would think there should be no change. The reason for there being a difference is that the MATLAB function `rgb2gray` has a standard weighting of $0.2989R + 0.5870G + 0.1140B$. These values were chosen as they are "visually pleasing" and closer to how the

Video	alpha			Accuracy old	Accuracy new
	R	G	B		
summeropen	0.1	0.7	1	58.33	77.96
mountainsun	0.1	0.1	1	63.67	75.12
summerdarkforest	0.1	0.3	0.9	77.93	81.45
rainforest	0.1	0.1	0.9	74.31	75.20
mrainforest	0.5	0.4	1	76.27	76.55
rainbuild	1	1	1	72.80	76.07
darkopen	1	1	1	72.53	77.75

Table 5.1: Comparison between weighted and non-weighted color components of the frames shown in Figure 3.2.



(a) regular grayscale at $[0.30, 0.59, 0.11]$

(b) weighted grayscale at $[1, 1, 1]$

Figure 5.3: Difference between regular and weighted grayscale

human eye interprets the different color components naturally [24]. The effect of this can be seen in Figure 5.3 where the regular grayscale looks more natural and makes it easier for a human to see details, while the weighted frame looks overexposed, but is easier for the computer to interpret and understand.

5.3 Finding threshold

The Otsu's segmentation method works better on each video, when they each have a separate threshold scale factor. The value of this scale factor heavily depends on weather and environment type in the videos. An example of how it looks when all the videos use the same scale factor can be seen in Figure 3.3 in the verification chapter.

To find the scale factor for each environment a MATLAB script was written. It compares the Otsu’s segmentation with different threshold scale factors and returns the best one. This is then done for all the environment types represented in the videos. The results are shown in Table 5.2. This method might not find the optimal threshold scale factor for each type of environment as only a few videos have been tested from each environment type and only a few frames from each video, but it will give a good indication of where an optimal threshold might be placed. This gives a better result than just setting the same threshold scale factor for all the videos regardless of weather conditions or surroundings in that particular video.

Most of the thresholds lie around 0.75 which coincides with Bergerskogens findings of `Autothreshold=0.3` being the best threshold in the summer type video he used [2].

Video	Threshold scale factor	Accuracy
summeropen	0.53	80.34
mountainsun	0.74	69.01
summerdarkforest	0.78	81.58
rainforest	0.81	75.12
mrainforest	0.76	76.09
rainbuild	0.65	76.53
darkopen	0.50	82.30

Table 5.2: Threshold scale factor for best segmentation using Otsu’s thresholding technique

5.4 Statistical Region Merging

Dhouha Attia et.al. did a comparison between different clustering algorithms in a very similar study to the one described in this thesis and concluded that Statistical Region Merging (SRM) seems to be the most promising algorithm for segmenting sky and non-sky [9]. The theory behind it is described in chapter 2.1.4. To confirm this it was compared to Otsu’s method which was the most promising in the reports from [4] and [2]. The results are shown in Table 5.3

As mentioned in the theory chapter, SRM is based on a statistical approximation heavily influenced by each pixel’s position. This gives it an advantage in avoiding misinterpreting small dots of sky as vegetation and vice versa, but it also means that when a cluster is wrongly classified, the whole cluster is classified wrong. So instead of just a couple of branches in a tree being classified as sky, the whole tree will be. The difference can be seen in Figure 5.4. Both are analyzed from the same frame, but with different segmenting methods. Otsu manages to catch each branch, but becomes jagged and finds a lot of holes of ”sky” in the vegetation. It has also failed to capture the mountain on the lower left edge. SRM is a lot smoother and manages to correctly see the mountain as non-sky. It has however completely missed two trees and included a cloud



(a) SRM overlay grayscale



(b) Otsu overlay grayscale

Figure 5.4: Image with overlay

As can be seen in table 5.3 the two are very similar in overall performance, but SRM works slightly better than Otsu on the videos with rain. The reason that Otsu’s method still was chosen over SRM is the same argument that Attia et.al. concluded with; It takes too much time, and the gain in accuracy is not worth it. Over all SRM spends over twice as much time as Otsu.

Data Set	Otsu		SRM	
	Overall acc.	Time	Overall acc	Time
summeropen	95.76	42.99s	98.58	159.96s
mountainsun	97.51	45.35s	97.81	131.58s
summerdarkforest	97.72	45.94s	94.67	140.58s
rainforest	96.55	46.96s	96.86	137.39s
mrainforest	97.83	50.08s	98.70	129.27s
rainbuild	96.99	48.19s	99.35	136.87s
darkopen	87.98	44.30s	98.45	142.55s

Table 5.3: Comparison between SRM and Otsu

5.5 Optimization

As mentioned in block 4.2.3 in the Proposed System chapter, the system can be made more effective and less time consuming by utilizing the fact that we are segmenting videos where neighboring frames are very similar.

There are two main choices to be made in trying to make the system more effective. Firstly a method has to be chosen to find the area in which the in-between frames should be segmented. This area is from here on called the *delta area*. Secondly a choice has to be made on the distance between each frame that is segmented as a whole. Testing has been done on both of these choices, and the results are given in the next two subsections.

5.5.1 Finding segment area

There are two main ways of finding the delta area. One is to separate out a block at the right elevation angles. The other is to find the difference between two frames and dilate it. Figure 5.5 shows the difference between two neighboring frames and the elevation angle lines. The lines show that most change happens between angles of 10-50 degrees.

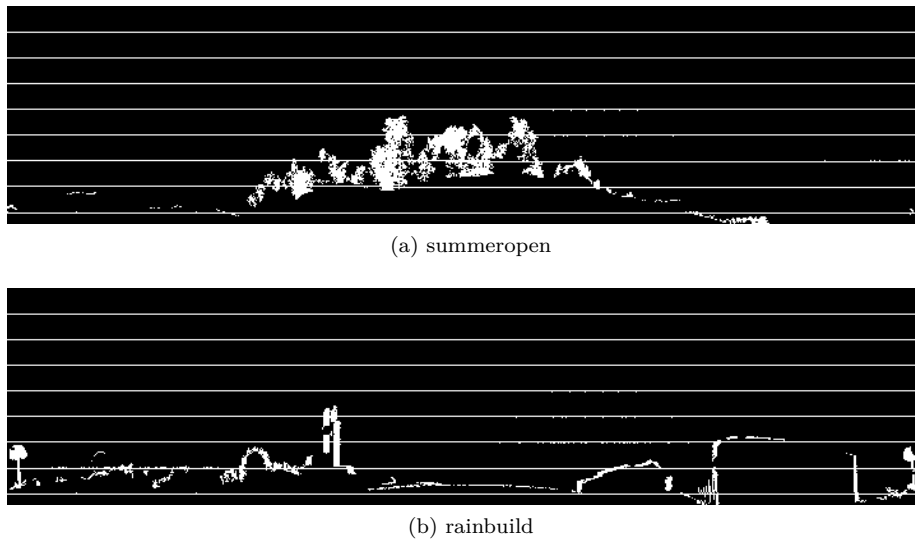


Figure 5.5: Difference between two frames with elevation angle lines from 10-90 degrees

Rectangular delta area

With change happening between 10 and 50 degrees, that is a block of 40 degrees that can be extracted and segmented by itself. This method is crude as it segments a lot more than it needs to and spends excess time doing this, but at the same time the step of finding a more fitting delta area is skipped, making this method less time-consuming in that department.

A drawback to this shape of the delta area is segmentation of tall structures. If an object extends beyond the borders of the delta area it will be cut off. This means that if a tall tree comes in to view, the first frame will show the whole

tree, but the second to fourth frames will show the top of the tree frozen from the first frame and the middle of the tree moving along the line without the top as seen with the structure in Figure 5.6a. The only way to cope with this is to make the block taller to capture more of the sky and potential tall structures. This in turn will make the algorithm as a whole slower and, if taken far enough, eliminate the benefit of the delta area as a whole.



(a) Structure being cut outside block range



(b) structure being cut outside dilated area

Figure 5.6: Flaws with the two shapes of delta area.

Dilated delta area

Instead of extracting the whole block, it is also possible to dilate the difference between the two last frames to create a delta area as shown in Figure 5.7. This method takes more initial time because of the dilation, but will spend less time in the segmentation part.

A drawback to this method is the capability to cope with sudden change. If a sudden cluster of trees come in to view, the algorithm will only segment them within the delta area as seen in Figure 5.6 until the delta area update where the whole cluster of trees is found. This problem will increase with extended increment time. There are two ways to lessen the effect of this flaw; decreasing the increment size or increasing the dilation size. Decreasing the increment size will make the algorithm slower over all because more frames throughout the video will have to be segmented fully. Increasing the dilation size will make the delta area bigger, thus catching more of the sudden changes, but this will again mean a bigger area will have to be segmented, spending more time on each frame. Over all it will be a compromise between required accuracy and time consumption.

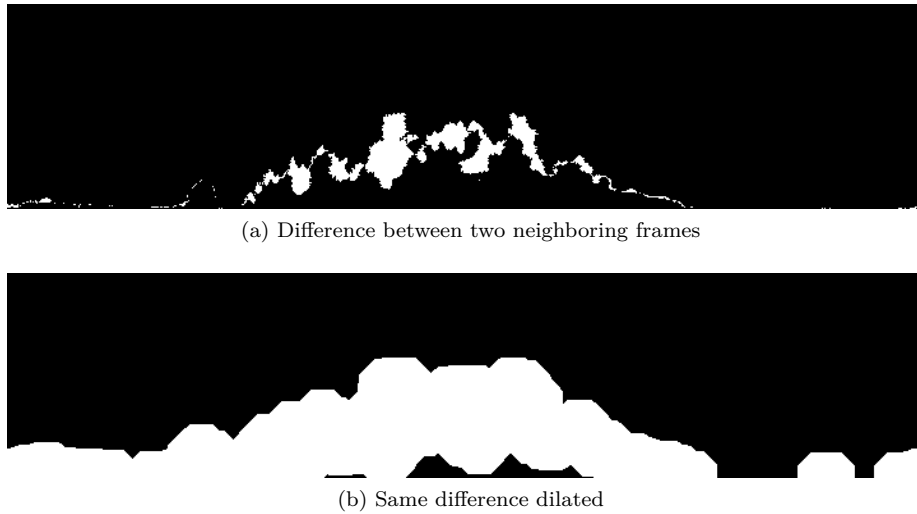


Figure 5.7: Dilated difference to create delta area

Both methods give good results, and as shown in Figure 5.6 they both have their flaws. Table 5.4 shows how the two compare in accuracy and time spent in segmentation. They are similar in accuracy with the dilated being marginally better. With regards to time consumption, the rectangular is close to 20 seconds faster for every minute of video. Despite this the dilated delta area is chosen because it through visual inspection showed to have a higher accuracy outside the sampled frames as well as a higher adaptability if an unexpected structure were to show up. The segmented videos using the two delta areas are compared side by side in a video in Appendix A.3.

Data Set	Dilated		Rectangular	
	Accuracy	Timing	Accuracy	Timing
summeropen	98.25	47.26s	95.57	32.65s
mountainsun	97.98	45.45s	97.98	28.37s
summerdarkforest	97.83	47.24s	97.83	31.31s
rainforest	98.15	49.78s	95.95	32.21s
mrainforest	96.64	45.79s	96.64	30.50s
rainbuild	98.35	52.83s	92.77	32.57s
darkopen	99.54	44.00s	78.75	31.67s

Table 5.4: Comparison between rectangular and dilated delta areas

5.5.2 Choice of Increment size

Increment size is the number of frames between each frame that is segmented as a whole. for example in the increment is 5, frames 1, 6 and 11 are segmented fully, while frames 2,3,4,5,7,8,9 and 10 are only segmented within the delta area and placed on top of frames 1 and 6 respectively. In general the accuracy of the segmentation system as a whole is better when the increment size is 1 i.e. when

each frame is segmented as a whole. It will also be more resilient towards sudden changes such as tall structures close to the road because the frame gets updated as whole more often. The compilation time on the other hand gets lower when the increment size gets bigger. All in all it will be a compromise between the time spent processing, and the resilience and accuracy of the segmentation.

The chosen increment size is 5. This number has been chosen because it allows for the computation time to be lowered at the same time as the frames stay comparatively similar. The shorter increment size will also give the system a better adaptability towards sudden changes without being overly time consuming. A future possibility is to make the choice of increment size be dynamic. With velocity data taken into account, the increment size can change depending on how fast the surroundings are changing. When the vehicle is driving fast, a short increment size is needed to cope with all the rapid changes, while it can be longer when the vehicle is driving slowly or standing still.

5.5.3 Dropping frames

The reference frames are numbers 1, 50, 100 and 150. All of these except the first are multiples of 5. With an increment size of 5, reference frames 50-150 will always be number 4 in the row, i.e. the last frame before the delta area is updated. To find a more accurate description of the accuracy, the first frames were dropped to see how the accuracy of the measured frames would change when they were number 1, 2, 3 and 4 in line after a fully segmented frame. The results can be seen in Table 5.5 and it seems like what order the sample frames are in does not impact the accuracy in any substantial way. This is good as it shows that the performance of the segmentation algorithm is not degraded by the use of a delta area. On the other hand it could suggest that the increment size can be increased further without decreasing the performance of the algorithm. This will in turn decrease the computation time.

Data Set	Accuracy at number of dropped frames				
	1	2	3	4	0
summeropen	98.25	98.00	98.13	97.59	95.47
mountainsun	90.14	96.96	91.18	87.78	97.93
summerdarkforest	93.22	93.80	97.34	98.08	98.01
rainforest	98.15	98.21	98.28	98.26	97.05
mrainforest	97.75	97.79	97.80	97.82	97.35
rainbuild	98.35	98.58	98.56	98.57	92.77
darkopen	99.54	99.17	99.54	99.53	92.18

Table 5.5: Accuracy as frames are dropped

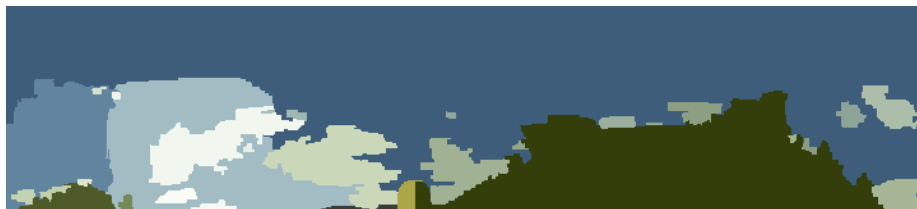
5.6 Three states

One of the objectives of this thesis was to separate the videos into three states: sky, vegetation and buildings/terrain. This has been challenging for a number of reasons. Most prominent of these is that most of the videos are recorded in areas with a lot of vegetation and the solid objects being vegetated mountains close to or far from the road. Vegetated mountains bear many similarities to vegetation, including color and outline, making a separation between them difficult to ascertain. In some parts of the video it is even difficult for humans to see the difference or the border line between them. A few methods were tested in trying to segment the videos into three states. These were SRM, Houghlines and Otsu's multilevel thresholding.

5.6.1 Statistical Region Merging



(a) SRM of video with buildings and rain



(b) SRM of video with mountain

Figure 5.8: Images showing the difficulties with finding three segments with SRM

SRM gives good results for separating LOS and NLOS, but it spends a lot of time doing this, making Otsu’s thresholding a better option. However, SRM is based on color image segmentation which may make it more suitable for separating the NLOS segment further into vegetation and buildings or terrain.

This method has been tested, but has not given any good results. A problem with SRM is that it finds a lot of irrelevant segments. Especially in the dark and rainy videos; the algorithm finds raindrops and clouds in the sky as segments before it separates the segments of bigger structures along the ground such as buildings or trees. This means that the post processing of the segmented video will be a more difficult task. A possible solution might be to find a way to tell the algorithm to weight clusters along the ground higher and thus separating them at a lower threshold than the sky and cloud clusters are separated. SRM can not be used alone in finding segments, but needs another algorithm on top to group the several SRM segments together into three states: sky, vegetation and solid objects. Two original frames and the SRM segmented ones at $Q = 32$ are shown in Figure 5.8.

The SRM algorithm manages to separate the buildings from the rest in the top frame, however the trees along the bottom edge are put in the same segment as the buildings and antenna, which does not make this algorithm an improvement from the segmentation into LOS and NLOS. The bottom frame shows the same problem with the vegetated mountain being classified as vegetation and the sky getting divided into more segments while the NLOS segment stays approximately the same.

5.6.2 Houghlines



(a)



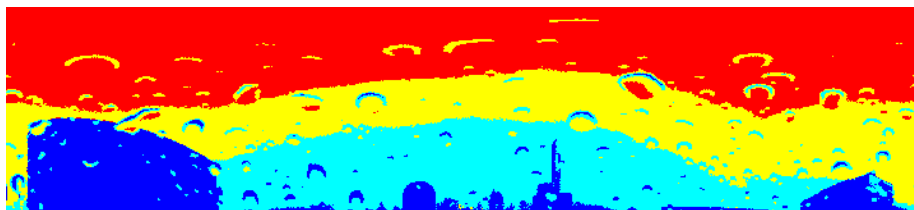
(b)

Figure 5.9: Houghlines on videos with buildings and mountain

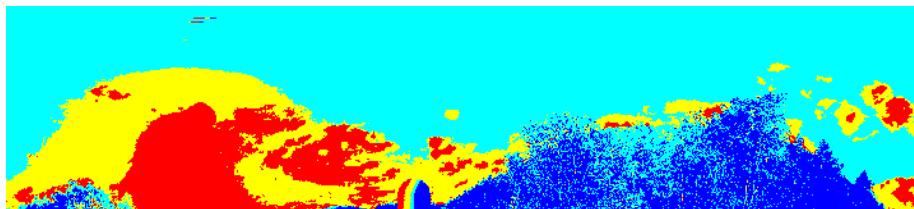
The Hough transform finds straight lines in an image. Since there are very few truly straight lines in nature, this was thought of as a plausible solution for

finding man made objects such as buildings and bridges and classifying them as solid. The first problem that arose was the fact that hemispherical lenses distort straight lines and make them curved. Since all the work in this thesis has been on hemispherical videos, this has been a challenge. In theory the curving should not happen with radial lines, therefore it was thought that Houghlines could find walls of houses if not the roofs. Surprisingly the opposite has been true for most of the frames. No good explanation has been found for this. The lines found are mostly short and even though they are in approximately the same place like seen on the left side of Figure 5.9.a, they will not be combined to form a single longer line. Figure 5.9.b shows a line found where there should be none. This has been a problem with the houghlines as well; the finding of random lines in vegetation an clouds. There is an option called Houghcircles that finds curves instead of straight lines, but this was not looked into.

5.6.3 Otsu's thresholding



(a) Otsu's multilevel threshold with buildings and rain



(b) Otsu's multilevel threshold with vegetated mountains

Figure 5.10: Original and segmented frames showing the difficulties with finding three segments using Otsu's multilevel thresholding. 3 thresholds give 4 segments

Otsu's thresholding has the possibility of setting multiple thresholds to segment out several parts of the image. This was used in trying to segment the video into three segments, but to no avail. The thresholding technique works well in separating sky and non-sky, but has a bigger problem when the intensity images are not as clean. Vegetation especially has a big variety of colors and intensities, making a thresholding technique very hard to use in discerning it as a whole from buildings or terrain. It might be a helpful preprocessing tool for enhancing texture, for example making the leaves on a tree stand out even more, but as a segmentation technique, it will not work in separating vegetation from buildings and terrain. A figure showing frames segmented with 3 Otsu's thresholds is shown in Figure 5.10.

In both of these frames the first threshold is approximately at the same value as the one used in segmenting the video into LOS and NLOS. The second and third thresholds are placed at higher values, meaning that they perform a better segmentation of the LOS segment, but does nothing with the one we are trying to segment further: the NLOS segment.

5.7 Statistical Model

Part of the problem formulation was to formulate a statistical model for the probability of blockage and attenuation at different elevation and azimuth angles relative to the vehicle and to cardinal directions. A simple algorithm has been made for this adding the segmented frames of the video together to construct a result image showing how many times a certain pixel has been marked as LOS. A collection of frames from the videos are shown in Figure 5.11.

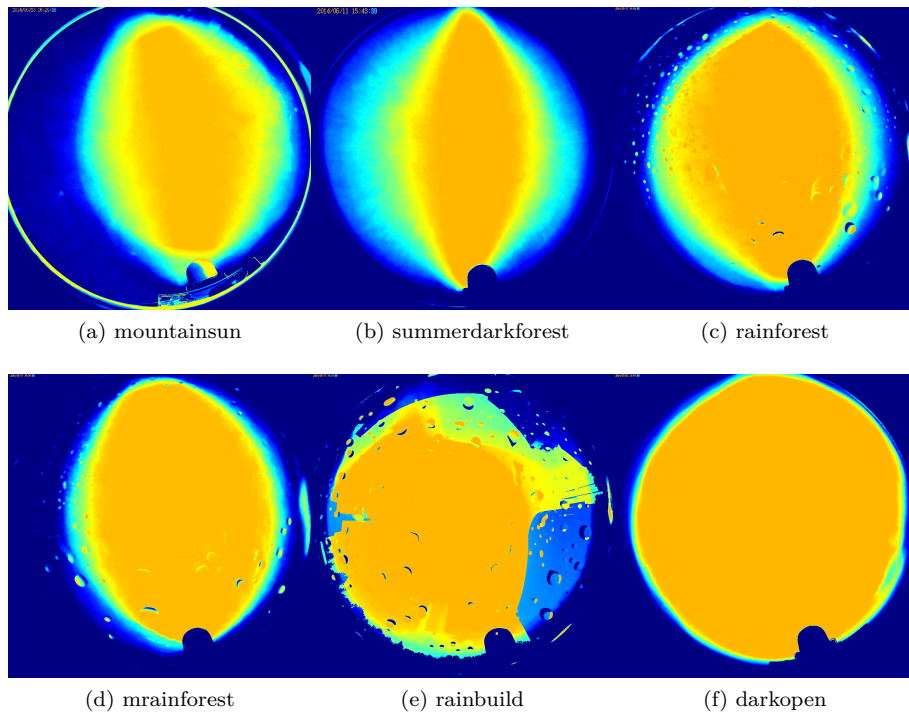


Figure 5.11: Probability of blockages at different azimuth and elevation angles in different environments

It becomes apparent that the probability of obstacles is highest at the sides of the car and lowest in front and behind as was also suggested by Rieche et.al. [8], though driving direction seemed to have less to do with the amount of blockage from the sides than other external factors such as mountains on the side of the road. The video from the urban area had a long period of the car standing still. This resulted in the clear overtraining of the particular area the car stood in.

The videos with raindrops on the lens that did not move throughout the video are also clearly tainted, but with a way to remove the raindrops from the segmentation, these images will be possible to use for analysis as well. The results fit well with what was found in the reports from Bråten et.al. [7] and Rieche et.al. [8].

The probabilities in relation to cardinal directions has not been found in a satisfactory way as the data set providing the vehicle's position and orientation does not belong to the analyzed videos but to an earlier data set. Even so an algorithm has been made for finding the probability of blockage as a function of cardinal directions using the dummy positioning data set. As of now this model will have no significance, but when combined with the correct positioning data, the algorithm can be used to find the probabilities of blockage as a function of elevation angles and cardinal direction in a similar way to what happens in the polar convert and rotation blocks of the proposed system.

The probability of line of sight at azimuths 0, 90, 180 and 270 degrees relative to the vehicle are shown in Figure 5.12. The results are similar to what one can read from the images in Figure 5.11 with the 0 and 180 degree graphs reaching a probability of close to 1 at lower elevations than the 90 and 270 degree graphs. This is especially visible in the *summerdarkforrest* video where the trees at the side of the road are tall, but the road is straight enough that the trees appear shorter due to perspective in the far front and rear of the vehicle. This makes the 0 and 180 degree graphs cluster together and the 90 and 270 degree graphs cluster together. In the *mountain_sun* video where there is a mountain on the left side of the video, the 270 degree graph grows more slowly towards 1 meaning that a satellite will need a higher altitude to go clear of the blockage provided by the mountain. on the left side of the vehicle than on the right.

From these plots it is possible to find how high a satellite needs to be in the sky to go clear of the blockage by a certain percentage. A realistic goal is to have a satellite be in clear view 95% of the time. In the *darkopen* video, this is possible at approximately 45 degrees elevation in all directions, while in the *summerdarkforrest* video this will take close to 80 degrees for an azimuthal position on the sides of the vehicle but only 17 degrees for a position in the front or back. Driving direction will have an influence on the required placement of the satellite when calculating the probability of blockage at azimuth angles in relation to the vehicle. When the positioning data belonging to the videos is applied, a more general model of probability for cardinal directions can be made that will give a better generalized view of the possible system requirements in a certain environment rather than the driving direction deciding everything.

The statistical model will also be further improved by the addition of a separation between attenuating and blocking objects in the NLOS segment.

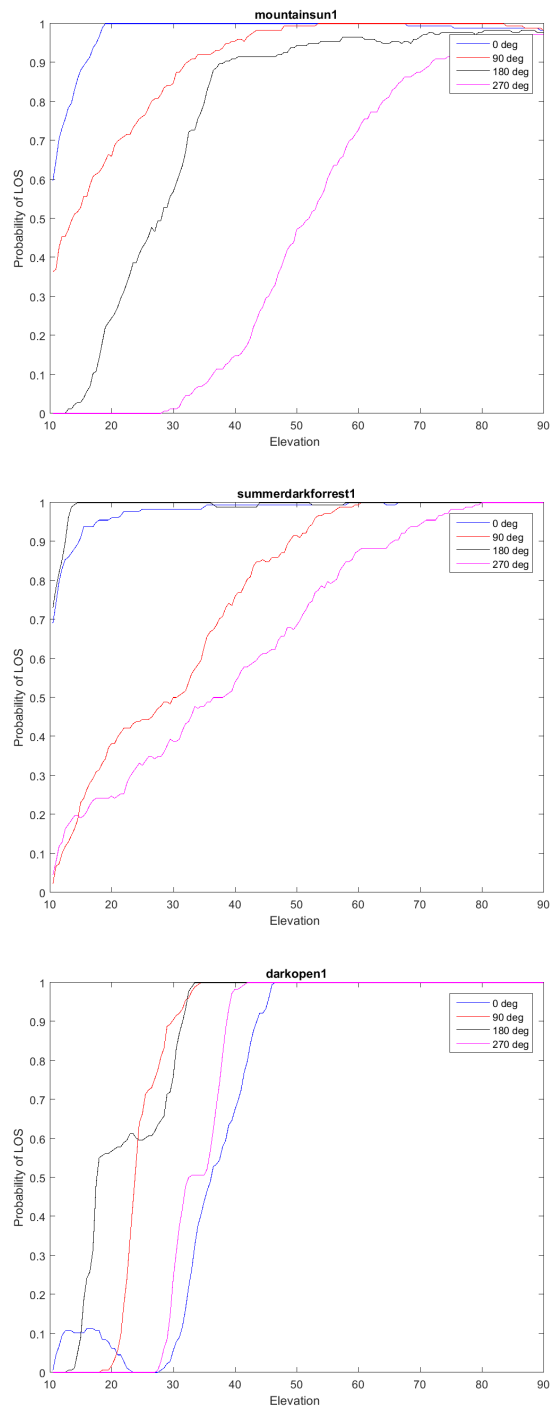


Figure 5.12: Probability of LOS at different elevations with azimuth 0, 90, 180 and 270 degrees in different environments

6. Discussion

6.1 Limitations

As explained in section 5.1 in the Experiments and Results chapter, the work was done on shortened versions of the original videos to make the file size and computation time manageable. This should not affect the performance of the proposed system, but might have been limiting in some way or another as it inadvertently has limited the number of different environments analyzed and tested.

Other possible limitations is that the small changes done to improve accuracy in one environment might affect the accuracy in other environments negatively. For instance may an accuracy improvement to a rainy video have a negative impact on the performance in a clear sky video. Ensuring that alterations to the algorithms have an overall accuracy and/or performance improvement would require a large dataset to be analyzed. And as the analysis of the provided data set takes hours to process, this was not practically possible.

6.1.1 Accuracy measurement

The overall accuracy of the system has been very good at 90% to 99%. It is however calculated solely from the frames that have a manually segmented reference image. This means that all the mentioned accuracies are based on 4 samples from each video. This is arguably not enough to say that one method is better than the other with full certainty, but it will give an indication as to what values overall will give the best results. Because of this, videos have been made that place the segmented video as an overlay over the original one to manually watch and get an indication of the actual accuracy. This too will just be an indication, though a better one as more of the video is taken into consideration outside of the 4 frames.

Another possible source of misinformation in the accuracy is the difference in proportions between the hemispherical image and the polar converted image. When measuring accuracy, the correctly classified pixels are divided by the total amount of pixels. This number will be skewed between the different representations because the polar converted image is interpolated along the top i.e. the middle of the sky. This makes the sky proportionally larger than the surroundings in this representation. The middle of the sky is easier to segment correctly making the proportion of correctly segmented pixels in the polar repre-

segmentation larger than in the hemispherical. This might be why the performance of the segmentation algorithm is close to 99% in the polar frames and closer to 80% when tested on the hemispherical frames.

6.2 Objective achievement

The main objectives of the thesis were to verify and improve the earlier system, segment the videos into three states, and make a statistical model for probable state at various elevation and azimuth angles both in two and three states.

Objectives 1 and 2 were completed with success; The verification proved to work as expected, and the improved system manages to segment the videos in two with a high success rate. In addition to this, there has been a big improvement in computation time. The third objective was completed with partial success; A statistical model has been made, and it works for azimuth and elevation relative to the vehicle. The extension to find probable state relative to cardinal directions is ready, but the actual data input is not available, and thus this objective is only partially completed.

Objectives 4 and 5 were not completed according to the assignment and problem formulation. This is due to difficulties in finding a method that worked reliably in separating the NLOS segment further into vegetation and buildings or terrain. A further explanation and demonstration of this is found in section 5.6 in the Experiments and Results chapter. Some more methods that seem promising to work further with, and possible improvements on the methods tested are described in section 6.4.

Because the fourth objective of segmenting the video into three states was not achieved, the fifth objective of constructing a model for probabilities of the three states as a function of azimuth and elevation angles in different environments was not achieved completely either. However, the basis for this statistical model is already laid through the model for probabilities of two states at various azimuth and elevation angles. As soon as a complete segmentation of the three states in the video is made, the already existing statistical model should be easy to expand, though an analysis of the model and the repercussions it will have on satellite communications will still need some work.

6.3 Computation Time

Through verifying Bergerskogen's work, his segmentation system was run on videos from the new data set. The times were around 110 seconds for a minute-long video with 90 seconds being the actual segmentation. With the system proposed in this thesis the corresponding time was around 70 seconds with 50 seconds of actual segmentation. That is an improvement of 45% in segmentation time and 36% in over all computation time, being a substantial improvement. With this improvement in computation time it is feasible to apply the system in realtime.

All these times are based on an assumption that the sample frequency is the same as the frame rate. This is seen as a reasonable assumption because processing the same frame twice yields the same segmentation result, meaning the only change between the samples will be the satellite's position in relation to the vehicle. This change in position will be negligible with the vehicle driving at 50km/h and the video already having a framerate of 3fps.

6.4 Future possibilities

6.4.1 Improving the current system

The biggest contribution to the work in this thesis would be to verify it. As mentioned in the Limitations section above, the accuracy measurements are not that reliable as they are done on only 4 samples throughout each video. Making more manually segmented reference images to check the accuracy of the system against is not a difficult task, but it is very time consuming. This is why the automated system is proposed in the first place; to relieve humans of the tiring task of having to go through whole videos and finding the state around the satellite in each frame.

Another big contribution would be to substitute the dummy log file for the real one to improve the statistical model for the probabilities of attenuation or blockage as a function of azimuth and elevation angles. With the real log data, the model can be expanded to take into account the azimuth and elevation angles in relation to the cardinal directions and not just to the local driving direction of the vehicle.

6.4.2 Making the system more applicable

Another way to improve the current system is to find a way to remove the antenna from the video frames. It has been attempted through a basic mask that filters out both the camera casing and antenna, but the antenna seems to move between videos. Supposedly this is because it was close to falling off the vehicle at one point during the measuring campaign and was mounted back on a little off where it used to be. If this is the case it might be at a more consistent place through videos in possible future measuring campaigns making it easier to remove it or mark it for further segmentation. Applying a mask to remove it is certainly the easiest way to avoid the bright yellow antenna to be classified as LOS, but a more useful way would be for the user to be able to click on it to mark where it is. A "click-to-mark" method could be used in simply removing the antenna and regarding it as a blind spot, but an even better solution still would be to try to estimate what is going to be behind this blind spot using the frames before and after in the video to see what objects "disappear" behind it. The blind spot is after all only present in the videos and will not be seen by the antenna, so having a big black spot will be a disadvantage.

This "click-to-mark" method would make the system more applicable to other measuring campaigns using other types of cameras or antennas, but it would require there to be made a GUI to be placed on top of at least parts of the

system. This in itself would make the system a lot more user friendly and other parts of the system could also benefit from it. The most prominent of which is the condition matrix. The way it is now, the user will be prompted in the MATLAB command window to enter a time. The time is then found in the condition matrix and the corresponding state (0 or 1) is returned. Making a visual layout would not only improve the overall user experience, but also make it easier to find the state at longer intervals of time, or go back to view the frames in question.

Another detail that would also make the system more applicable to other measuring campaigns with different cameras is to find the center of the camera lens automatically. This was also attempted in this work by looking at how the camera housing intersects with the edges of the frame and trying to ascertain the curvature to be able to formulate a circle from it and from this finding the center of the lens. It did not work as expected, and the provisional solution of counting how many pixels off center the circle is was used in the final product. The curvature to circle to center method might work given more time and effort.

6.4.3 Motion tracking

A possibly big improvement in the performance of the system is tracking objects as the vehicle drives by them. This has been a challenge in most of the videos in the data set as the frame rate is too low relative to the speed of the vehicle to find objects to track. As a human watching the video, we can follow some of the objects as the vehicle drives by, but with three frames every second, it is difficult for the computer to recognize a tree near the top in one frame and a tree near the bottom in the next frame as the same object. This may be solved by increasing the frame rate of the original video. It may also be solved by going deeper into the subject of motion tracking or branches thereof to find a method that could work even in lower frame rates.

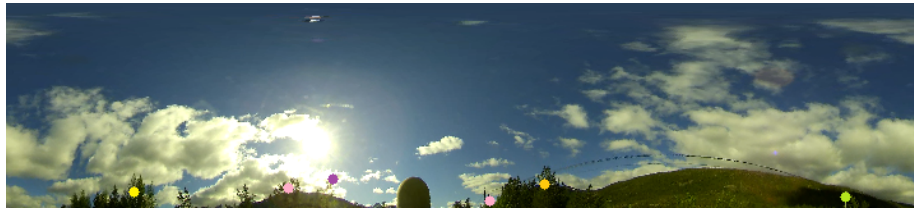
Making motion tracking work would present an opportunity to make the video segmentation a lot smoother. Especially lampposts have a tendency to be lost when getting closer to the horizon, making it look like they pop in and out of existence in the segmented video. Tracking the lampposts and estimating where they should be in the next frame will make it easier for the algorithm to find and correctly classify them as hard objects. It is possible in film making to track objects and simulate perspective so they are recognized as the same object even when seen from another angle. The method behind this might work on the data set.

6.4.4 Three states

A big part of any future work will be to separate the NLOS segment into vegetation and hard objects. A few methods have been tried in the work with this thesis, but no good solutions have been found. However, a few paths have shown promise. These are described in the next subsections. Most of the videos in the data set were recorded in rural and highly vegetated areas. So the solid objects in the video are mostly terrain, not man made buildings or bridges. This



(a) Frame 1 with selected tracking dots



(b) Frame 2 with selected tracking dots

Figure 6.1: Tracking of selected dots

makes a texture based segmentation more applicable than an algorithm looking for man made structures as there are only a few of these in the data set.

Texture

A big possibility in any future work is to look at texture to separate the NLOS segment further into attenuating (vegetation) and blocking objects (buildings, mountains etc.). Time constraints did not allow it to be thoroughly tested in this work, but based on literature studies, it seems like a viable option. Especially Meurie et.al. in [10] did a good study in of how it might work specifically in a LOS/NLOS setting. Using a texture approach to segmenting the last two segments will require a database of sample textures to train the classifier on. The Berkeley Segmentation Dataset Benchmark (BSD3) is a frequently used database for this kind of work. It consists of over 12,000 segmentations of 1,000 Corel dataset images and is free to download for non-commercial research and education purposes, provided that the developed segmentation algorithm is shared back with them *'in the spirit of cooperative scientific progress'*(Martin 2001 [25]).

To use this database would massively lighten the work of anyone trying to make a texture database. However, the algorithm will still need to be trained to correctly separate vegetation and solid objects. Because there is more prominent texture in vegetation than in buildings, this is a good place to start separating them. There are very few flat and patternless areas in nature, so any objects like this is very likely to be man made and classified as solid. Classifying the areas that have the highest confidence levels first will make the algorithm more effective as less parts of the frame will have to be reanalyzed or inspected further.

The texture information can also be combined with a probability function saying the likelihood of different states at different elevation angles to further aid the classification algorithm in deciding if an object is vegetation or solid.

Segmenting based on texture will also give an opportunity to differentiate between heavy and light vegetation if this is desirable.

Statistical Region Merging

SRM has given good results for separating LOS and NLOS, though it is too time consuming to be a justifiable solution. In separating the NLOS segment further into vegetation and buildings or terrain it has shown promise, but due to time constraints, a good solution was not found. SRM will most likely not be a viable solution on its own, but in combinations with other methods such as texture or alternative color spaces, it might be a good tool. In this thesis it has also only been tested on dark rainy environments and vegetated mountains. As it is a segmentation technique based on color, it might do better in environments where the objects differ more in color such as an urban or suburban area in better light than what was present in the videos of the current data set. Anyone doing further work on this must be prepared that SRM is a time consuming method, though it gives good and stable results. In a further segmentation of the NLOS segment it might be worth the extra time spent processing.

Statistics

To use statistical models for finding the probabilities of each segment at different points in the frame would be a big help in finding the different types of objects in the videos. Such models can be found as recommendations from ITU-R [3]. These are mostly based on satellite communication closer to the subsatellite point, but might still be of help

The main goal of this thesis has been to make such a model for satellite communication far north to be a basis for recommendations on positioning of satellites in non-geostationary SATCOM. Even so, existing models will give an indication as to what one might expect in different environments and at different azimuth and elevation angles. If combined with a trainable classifier, this information could be transformed or at least added to other kinds of information to help decide the state of segments found in the videos.

7. Conclusion

This thesis has proposed a system for segmenting a hemispherical video into LOS and NLOS segments. It is an improvement on earlier works by Bergerskogen [2] and Ruud et.al.[4]. The proposed system works by taking in a hemispherical video and polar converting it frame by frame. The frames are then segmented using a combination of GRD and Otsu's thresholding method. This step has been made more efficient by only segmenting the areas that change between frames. Every 5th frame is segmented fully while the 4 frames in between are segmented inside a delta area based on the first frame. The segmented frames are saved in a condition matrix to give the user the option of going back to check the state at certain times and positions, and as a video for manually checking approximate accuracy. The accuracy lies between 90% and 99% in the tests that have been run, but through visual inspection of the segmented video, it becomes clear that these tests are a mere indication. The actual accuracy is a bit lower, but is still an improvement of the earlier system based on visual comparisons and accuracy perception. It cannot be guaranteed that the proposed system has improved segmentation accuracy without proper evidence, but the proposed system does give an improvement in segmenting time of around 45%. This is such an improvement that a realtime application could be considered.

A statistical model for the probabilities of blockage as a function of azimuth and elevation angles relative to the vehicle was also made. In general it is more likely for a satellite to be within line of sight in front of and behind the vehicle, and less likely at the sides. This model can be useful in setting system requirements for the satellite's position in future SOTM projects or SATCOM with non-geostationary satellites. The model is expandable as information on the vehicle's position in relation to cardinal directions is obtained, and as further information on vegetation and solid objects within the NLOS segment is found.





Not all of the objectives of the thesis were achieved. A separation of the NLOS segment into attenuating and blocking objects proved too difficult within the time constraints. This will be the most prominent part of any future studies done by the author on this subject. A success in this will mean an improvement in the statistical model making it able to factor in attenuation as opposed to blockage in the probabilities for good signal quality as a function of azimuth and elevation angles. The basis for this model is already made through the statistical model for two states.

Abbreviations

- AVI - Audio Video Interleaver
- BSDB - Berkeley Segmentation Dataset Benchmark
- CAPLOC - Combinaison de l'Analyse d'image et de la connaissance de la Propagation des signaux pour la Localisation
- DLOS - Direct Line Of Sight
- FFI - Forsvarets Forskningsinstitutt (Norwegian Defence Research Establishment)
- GNSS - Global Navigation Satellite System
- GPS - Global Positioning System
- GRD - Geodesic Reconstruction by Dilation
- ITU-R - International Telecommunication Union's Radiocommunication sector
- LOS - Line Of Sight
- MPEG-4 - Motion Picture Experts Group Layer-4 Video
- MiLADY - Mobile satellite channel with angle diversity
- NLOS - Not Line Of Sight
- PoE - Power over Ethernet
- SATCOM - Satellite communication
- SHF - Super High Frequency
- SNR - Signal to Noise Ratio
- SOTM - SATCOM on the move
- SRM - Statistical Region Merging
- UIS - Universitetet i Stavanger (University of Stavanger)
- VLC - Video LAN Client

A. Appendices

click the paperclips to open the appendix files

-  **A.1 Original Problem formulation**
-  **A.2 Segmented video with overlay**
-  **A.3 Video showing delta areas for block and dilation**
-  **A.4 MATLAB code for Video analyzer .zip**

Bibliography

- [1] IIS F. Milady - mobile satellite channel with angle diversity, 2012. URL <https://artes.esa.int/projects/milady-mobile-satellite-channel-angle-diversity>.
- [2] Bergskogen P. Klassifisering og forbedring av metode for segmentering av video fra fisheye-kamera. Technical report, Norwegian Defence Research Establishment, July 2014.
- [3] ITU-R. Attenuation in vegetation. Recommendation P.833-7, International Telecommunication Union, 02 2012.
- [4] Ruud AM, Skoglund R, Wedervang TM. Videoanalyse av fisheyelinse i forbindelse med mobil satelittkommunikasjonr. Bachelor's thesis, Gjøvik University College, May 2014.
- [5] Arndt D, Heyn T, Heuberger A, Prieto-Cerdeira R, Eberlein E. State modeling of the land mobile satellite channel with angle diversity. In 2012 6th European Conference on Antennas and Propagation (EUCAP). ISSN 2164-3342, March 2012; 3140–3144.
- [6] Marais J, Tay S, Flancquart A, Meurie C. Weighting with the pre-knowledge of gnss signal state of reception in urban areas. In European Navigation Conference 2015. 2015; .
- [7] Bråten LE, Amaya C, Rogers DV. Statistical characterization of land mobile-satellite propagation environments using a photogrammetric technique. International Union of Radio Science 2002;.
- [8] Rieche M, Arndt D, Ihlow A, Pérez-Fontán F, Galdo GD. Impact of driving direction on land mobile satellite channel modeling. In The 8th European Conference on Antennas and Propagation (EuCAP 2014). ISSN 2164-3342, April 2014; 2268–2271.
- [9] Attia D, Meurie C, Ruichek Y, Marais J. Counting of satellites with direct gnss signals using fisheye camera: A comparison of clustering algorithms. In Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on. ISSN 2153-0009, Oct 2011; 7–12.
- [10] Meurie C, Ruichek Y, Cohen A, Marais J. A hybrid and adaptive segmentation method using color and texture information, 2010. URL <http://dx.doi.org/10.1117/12.838923>.

- [11] VIVOTEK. Fisheye Network Camera, 2013.
- [12] MathWorks. Supported video file formats, 2015. URL http://se.mathworks.com/help/matlab/import_export/supported-video-file-formats.html.
- [13] MauriceSnell. Vlc support guide ,transcode, 2015. URL <https://wiki.videolan.org/Transcode/1>.
- [14] Otsu N. A threshold selection method from gray-level histograms. In Systems, Man and Cybernetics, 1979 9th IEEE Transactions on, volume 1. 1979; 62–66.
- [15] BDataID Your guide in the bar code jungle. What’s ocr. URL <http://www.dataid.com/aboutocr.htm>.
- [16] Milyaev S, Barinova O, Novikova T, Kohli P, Lempitsky V. Image binarization for end-to-end text understanding in natural images. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition, ICDAR '13. Washington, DC, USA: IEEE Computer Society. ISBN 978-0-7695-4999-6, 2013; 128–132. URL <http://dx.doi.org/10.1109/ICDAR.2013.33>.
- [17] Duda RO, Hart PE. Use of the hough transformation to detect lines and curves in pictures. Commun ACM January 1972;15(1):11–15. ISSN 0001-0782. URL <http://doi.acm.org/10.1145/361237.361242>.
- [18] Nock R, Nielsen F. Statistical region merging. IEEE Trans Pattern Anal Mach Intell November 2004;26(11):1452–1458. ISSN 0162-8828. URL <http://dx.doi.org/10.1109/TPAMI.2004.110>.
- [19] Boltz S. Image segmentation using statistical region merging, 2010. URL <http://se.mathworks.com/matlabcentral/fileexchange/25619-image-segmentation-using-statistical-region-merging>.
- [20] MathWorks. vision.autothresolder system object, apr 2016. URL <http://se.mathworks.com/help/vision/ref/vision.autothresolder-class.html>.
- [21] Satre JT. Satellite look angle calculator, 2012. URL <http://www.satellite-calculations.com/Satellite/lookangles.htm>.
- [22] Alan. Satellite finder / dish alignment calculator with google maps, 2012. URL <http://www.dishpointer.com/>.
- [23] Mitu C. Satlex calculator for azimuth and elevation angle, 2010. URL <http://www.satlex.de>.
- [24] Shoelson B. Mathworks file exchange , customgray, 2013. URL <http://www.mathworks.com/matlabcentral/fileexchange/43716-customgray/content/customGray.m>.
- [25] Martin D, Fowlkes C, Tal D, Malik J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proc. 8th Int’l Conf. Computer Vision, volume 2. July 2001; 416–423.