

Classification between vegetation and non-vegetation using hyperspectral sensing

Thesis submitted in partial fulfillment of the requirements for the M.Sc. degree

By: Vitzrabin Efraim

Advisors: Prof. Y. Edan, Dr. V. Alchanatis

Ben-Gurion University of the Negev, Faculty of Engineering Sciences
Department of Industrial Engineering and Management

January 2011

Classification between vegetation and non-vegetation using hyperspectral sensing

Thesis submitted in partial fulfillment of the requirements for the M.Sc. degree

By: Vitzrabin Efraim

Advisors: Prof. Y. Edan, Dr. V. Alchanatis

Ben-Gurion University of the Negev, Faculty of Engineering Sciences

Department of Industrial Engineering and Management

Author: Vitzrabin Efraim

Advisor: Prof. Edan Yael

Advisor: Dr. Alchanatis Victor

Chairman of graduate students committee: Prof. Kreimer Joseph

January 2011

Acknowledgments

This research would not have been possible without the support of many organisations and people.

I wish to express by deepest gratitude to my advisors, Prof. Yael Edan and Dr. Victor Alchanatis, who were most helpful, and without their insight, support, guidance and assistance, this thesis would not have been fulfilled.

I wish to thank Israel Aerospace Industries, LAHAV division who let me do this thesis while I was working there, supporting me with everything I needed.

Great thanks, goes to all of my coworkers at IAI, lead by Ofir Cohen and Asher Becher, who were more than just coworkers, but true friends that accompany and supported me through the entire period.

I wish to thank Volcani Institute, for letting me use their facilities and equipment as much as I needed.

Special thanks go to my closest friend Danny Eizicovits, for your insightful ideas and support, even when I bothered you in the middle of the night, and for the countless hours that we have talked and debated in relation to this thesis.

At last, I wish to thank my family and friends, for their moral support through the entire process of writing this thesis.

Abstract

Vegetation classification is very important for various applications like autonomous navigation and agricultural robots. In autonomous navigation, most obstacles detection algorithms are based on geometry and color analyses. Hence, soft obstacles like stems would be detected as obstacles and the vehicle would avoid them instead of driving through them, which causes unreasonable navigation. For many agricultural robotic applications, the vehicle must detect drivable paths in order not to destroy crops and must be able to distinguish between vegetation and non-vegetation. Usually paths are detected geometrically using laser sensors and color cameras. However, this is problematic in outdoor environments with changing light conditions (e.g., direct sun light, clouds ...). The ability to classify the material of the path, and not only its geometric shape or color can provide a better and more robust way to identify paths.

This research focused on classification between vegetation and non vegetation in outdoor conditions.

A hyperspectral imaging system was used to acquire wavelengths in the range of 500 to 900 nm with 5nm FWHM (full width half maximum intervals). Different scenes were captured under different light conditions (direct sunlight, clouds, and different times of the day). The scenes contained several types of vegetation as well as different objects, like asphalt, soil and building walls. Seven algorithms were developed and adapted for vegetation and non-vegetation classification and their performances were compared: 1) the Mahalanobis distance algorithm, compared the Mahalanobis distance of a pixel to a known vector of vegetation; 2) the Derivative algorithm calculated the difference between each two consecutive wavelengths and compared it to a known vegetation difference. Instead of using all of the derivatives, only the most influencing ones were used; they were selected by a “C5.0” decision tree algorithm; 3) the Spectral Angle Mapper algorithm considered the wavelengths of a pixel as an N dimensional vector and calculated the angle between the pixel and a known vegetation vector. 4) a variation of the Spectral Angle Mapper, which instead of using all the data used only the most important wavelengths. The most important wavelengths were chosen by a decision tree “C5.0” algorithm; 5) the Normalized Difference Vegetation Index (NDVI) algorithm which used two wavelengths, and calculated the normalized difference between them. For optimally choosing the parameters, a decision tree “C5.0” algorithm was used; 6) an extension of the NDVI algorithm: it calculated several NDVI's, and used a decision tree for the classification; 7) an adaption of the *NDVI* algorithm that considered the exposure time of the hyperspectral system, and searched for the best exposure time for each wavelength.

Developments included:

- Automatic selection of optimal parameters for the algorithms,
- Automatic selection of optimal wavelengths, and
- Three new algorithms (MSAM, NDVI_T and dynamic channel relation) based on existing algorithms.

The results indicated that the SAM and the NDVI algorithms were the best out of the seven algorithms, with a classification accuracy of over 97% and 96% respectively. The two algorithms were not affected by changes in light, which was one of the basic requirements.

Keywords: hyperspectral imaging, vegetation classification.

Table of Contents

| | |
|---|-----------|
| CHAPTER ONE: INTRODUCTION | 1 |
| 1.1 DESCRIPTION OF THE PROBLEM | 1 |
| 1.2 OBJECTIVE | 1 |
| 1.3 RESEARCH INNOVATIONS..... | 2 |
| CHAPTER TWO: LITERATURE REVIEW | 3 |
| 2.1 HYPERSPECTRAL SENSING..... | 3 |
| 2.2 INTERACTION BETWEEN THE SUN RADIATION AND AN OBJECT | 4 |
| 2.3 VEGETATION SPECTRAL BEHAVIOR | 5 |
| 2.4 GENERAL HYPERSPECTRAL ALGORITHMS | 6 |
| CHAPTER THREE: METHODOLOGY | 13 |
| 3.1 GENERAL | 13 |
| 3.2 EXPERIMENTAL SETUP | 13 |
| 3.3 ALGORITHMS..... | 16 |
| 3.4 PERFORMANCE MEASURES..... | 16 |
| 3.5 DECISION TREE..... | 18 |
| 3.6 CALIBRATION | 19 |
| 3.7 CLOUDS INTERFERENCE | 19 |
| 3.8 SENSITIVITY ANALYSIS..... | 20 |
| CHAPTER FOUR: ALGORITHMS..... | 21 |
| 4.1 OVERVIEW | 21 |
| 4.2 MAHALANOBIS DISTANCE..... | 21 |
| 4.3 DERIVATIVE ALGORITHM | 22 |
| 4.4 SPECTRAL ANGLE MAPPER (SAM)..... | 23 |
| 4.5 MINOR SPECTRAL ANGLE MAPPER (MSAM) | 24 |
| 4.6 CHANNEL RELATION (NDVI)..... | 24 |
| 4.7 CHANNEL RELATIONS TREE (NDVI_T) | 26 |
| 4.8 DYNAMIC CHANNEL RELATIONS | 27 |
| CHAPTER FIVE: RESULTS AND DISCUSSION | 28 |
| 5.1 MAHALANOBIS DISTANCE..... | 28 |
| 5.2 DERIVATIVE | 31 |
| 5.3 SPECTRAL ANGLE MAPPER (SAM)..... | 35 |
| 5.4 MINOR SPECTRAL ANGLE MAPPER (MSAM) | 38 |

| | |
|---|-----------|
| 5.5 CHANNEL RELATION (NDVI) | 42 |
| 5.6 CHANNEL RELATIONS TREE (NDVI_T)..... | 45 |
| 5.7 DYNAMIC CHANNEL RELATION | 49 |
| 5.8 CLOUD INTERFERENCE | 51 |
| 5.9 DISCUSSION | 52 |
| CHAPTER SIX: CONCLUSIONS AND FUTURE WORK | 53 |
| 6.1 CONCLUSIONS..... | 53 |
| 6.2 FUTURE WORK | 53 |
| CHAPTER SEVEN: REFERENCES | 54 |
| APPENDICES..... | 56 |
| APPENDIX A SOFTWARE | 56 |
| APPENDIX B STATISTICAL ANALYSIS..... | 68 |
| APPENDIX C DECISION TREES..... | 78 |
| APPENDIX D RAW DATA | 80 |
| APPENDIX E MANUALLY CLASSIFIED IMAGED | 80 |

LIST OF TABLES

| | |
|--|----|
| Table 1 – Summary of the advantages and disadvantages of the algorithms..... | 11 |
| Table 2 – Examples of algorithms in applications | 12 |
| Table 3 – Confusion Matrix | 17 |
| Table 4 – Mahalanobis - Independent evaluation of hypercubes | 28 |
| Table 5 – Mahalanobis – Cross Validation | 28 |
| Table 6 – Mahalanobis – Training by two hypercubes | 29 |
| Table 7 – Mahalanobis – Different light conditions..... | 30 |
| Table 8 – Mahalanobis – Noise | 30 |
| Table 9 – Mahalanobis – Training by different number of hypercubes | 31 |
| Table 10 – Derivative - Independent evaluation of hypercubes..... | 32 |
| Table 11 – Derivative – Cross Validation | 32 |
| Table 12 – Derivative – Training by two hypercubes | 33 |
| Table 13 – Derivative – Different light conditions | 33 |
| Table 14 – Derivative – Noise..... | 34 |
| Table 15 – Derivative – Training by different number of hypercubes..... | 34 |
| Table 16 – SAM - Independent evaluation of hypercubes | 35 |
| Table 17 – SAM – Cross Validation | 35 |
| Table 18 – SAM – Training by two hypercubes | 36 |
| Table 19 – SAM – Different light conditions..... | 37 |
| Table 20 – SAM – Noise | 37 |
| Table 21 – SAM – Training by different number of hypercubes | 38 |
| Table 22 – MSAM - Independent evaluation of hypercubes | 39 |
| Table 23 – MSAM – Cross Validation..... | 39 |
| Table 24 – MSAM – Training by two hypercubes..... | 40 |
| Table 25 – MSAM – Different light conditions | 41 |

| | |
|--|----|
| Table 26 – MSAM – Noise | 41 |
| Table 27 – MSAM – Training by different number of hypercubes | 42 |
| Table 28 – NDVI - Independent evaluation of hypercubes..... | 42 |
| Table 29 – NDVI – Cross Validation..... | 43 |
| Table 30 – NDVI – Training by two hypercubes | 43 |
| Table 31 – NDVI – Different light conditions | 44 |
| Table 32 – NDVI – Noise..... | 44 |
| Table 33 – NDVI – Training by different number of hypercubes..... | 45 |
| Table 34 – NDVI_T - Independent evaluation of hypercubes | 46 |
| Table 35 – NDVI_T – Cross Validation | 46 |
| Table 36 – NDVI_T – Training by two hypercubes | 47 |
| Table 37 – NDVI_T – Different light conditions..... | 48 |
| Table 38 – NDVI_T – Noise | 48 |
| Table 39 – NDVI_T – Training by different number of hypercubes | 49 |
| Table 40 – Comparison between the different algorithms | 52 |

List of Figures

| | |
|---|----|
| Figure 1 – Hypercube example (Short, 2009) | 3 |
| Figure 2 – Sun radiation at various places (Solar radiation handbook, 2008) | 4 |
| Figure 3 – Reflectance of a pine tree..... | 6 |
| Figure 4 – Reflectance of a petunia flower | 6 |
| Figure 5 – Hyperspectral imaging system architecture | 14 |
| Figure 6 – Photograph of the hyperspectral imaging system | 14 |
| Figure 7 – Different types of environments | 15 |
| Figure 8 – Example of an image and a manual classified image | 17 |
| Figure 9 – Derivative, vegetation decision tree..... | 23 |
| Figure 10 –Channel relation - vegetation decision tree..... | 25 |
| Figure 11 – Multiple channel relation - vegetation decision tree..... | 26 |
| Figure 12 – Example of Mahalanobis distance classification | 29 |
| Figure 13 – Example of Derivative classification | 33 |
| Figure 14 – Example of SAM classification | 36 |
| Figure 15 – Example of MSAM classification..... | 40 |
| Figure 16 – Example of NDVI classification | 43 |
| Figure 17 – Example of NDVI_T classification | 47 |
| Figure 18 – Total success in dynamic channel relation algorithm | 50 |
| Figure 19 - False positive rate in dynamic channel relation algorithm | 50 |
| Figure 20 - False negative rate in dynamic channel relation algorithm | 50 |
| Figure 21 – Normalized difference between direct sunlight and clouds | 51 |

Chapter One: Introduction

1.1 Description of the Problem

Autonomous ground vehicles that use current sensors technologies such as laser scanners, cameras, radars and ultrasonic sensors often bypass soft obstacles like long leaves and high vegetation. This is mostly due to the fact that current algorithms are based on geometric and color analyses and encounter problems in classifying materials (Fogler 2003). The failure to reliably distinguish between vegetation and non-vegetation is crucial for reasonable driving, to ensure the vehicle stays on the path. In agriculture it is more important not to drive on the vegetation because the crops are often parts of it.

Current systems for sensing vegetation include 3d laser sensors (Rutzinger et al., 2010), color cameras (Bradley et al., 2007), and multi and hyperspectral systems (Fogler 2003, Chen et al., 2010). Current research achievements in outdoor classification are limited to 90% accuracy (Bradley et al., 2007).

Hyperspectral sensing deals with acquiring and processing imaging for tracing materials and phenomenon on the surface. This imaging is usually comprised of dozens of narrow bands between the UV to the thermal infrared regions (0.4 to 14 μm). It is assumed that the object's physical and chemical characteristics correlate to the radiation that reflects from the object (a different amount is reflected at each wavelength depending on the material of the object, Landgrebe, 2002). Due to dozens of bands involved in the process, the problem dimension is very big, thus, sensitive to different kinds of noise that do not behave the same way in all of the bands. Since the hyperspectral imaging system is passive (it is a sensor that does not transmit any radiation to the outside world, only absorbs the radiation), the light source also has strong influence on its performance. When the light source is the sun, the atmosphere affects the behavior of the light on the ground; moreover, the object can be shaded by clouds or by other objects, which cause changes to light that hits it.

Utilizing a hyperspectral imaging system in order to classify vegetation can add a very important data source to an autonomous vehicle in order for it to achieve its goals successfully.

1.2 Objective

The research objective is to develop an algorithm to classify between vegetation and non- vegetation in changing outdoor light conditions using a hyperspectral sensor without any a-prior knowledge on the scene. The aim is to increase detection accuracy above 95%.

1.3 Research innovations

This research is based on existing algorithms that have been previously used in hyperspectral imaging. However, several new approaches have been developed:

- The algorithms use *only hyperspectral imaging* as opposed to other research which employed additional sensors like laser scanners (Bradley et al., 2007).
- *Optimal wavelengths* were *automatically selected*. Prior algorithms that dealt with hyperspectral imaging, selected wavelengths based on information from previous research and known behavior (Alchanatis et al., 2005). Plotting several pixels on a graph and selecting from there the best ones (Ye et al., 2008), or executing the same algorithm several times, each time with a different wavelength (Yang et al., 2003). To obtain the optimal wavelengths in this thesis, a decision tree approach was taken assuming no assumption for a unique statistical behavior (like normal distribution).
- The performance of every algorithm is highly dependent on its parameters. In this research *optimal parameters were derived* using a decision tree approach, instead of using a regression method (Bradley et al., 2007), or choosing them through experiments.
- *Three new algorithms were developed* (MSAM, NDVI_T and dynamic channel relation) by adapting existing algorithms.

Chapter Two: Literature Review

2.1 Hyperspectral Sensing

Hyperspectral sensing is the ability to acquire information from an area by separating the spectral dimension of the electromagnetic radiation returning from the area into narrow and continuous bands (Shaw et al., 2002). Nowadays, due to the improvement of computer power, many applications in various fields (e.g., agriculture, intelligence and environment monitoring) rely on hyperspectral sensing (Landgrebe, 2002, Antonucci et al., 2010).

Hypercube

During hyperspectral imaging a hypercube (Figure 1) is acquired. The hypercube is a three dimensional cube which has two spatial dimensions (x, y) and one spectral dimension (z). The value of each pixel in the two spatial dimensions is the returned electromagnetic radiation at that pixel. The z axis includes all the wavelengths images (i.e., an image for each wavelength; in the following example the hypercube includes 224 images, one image for each wavelength).

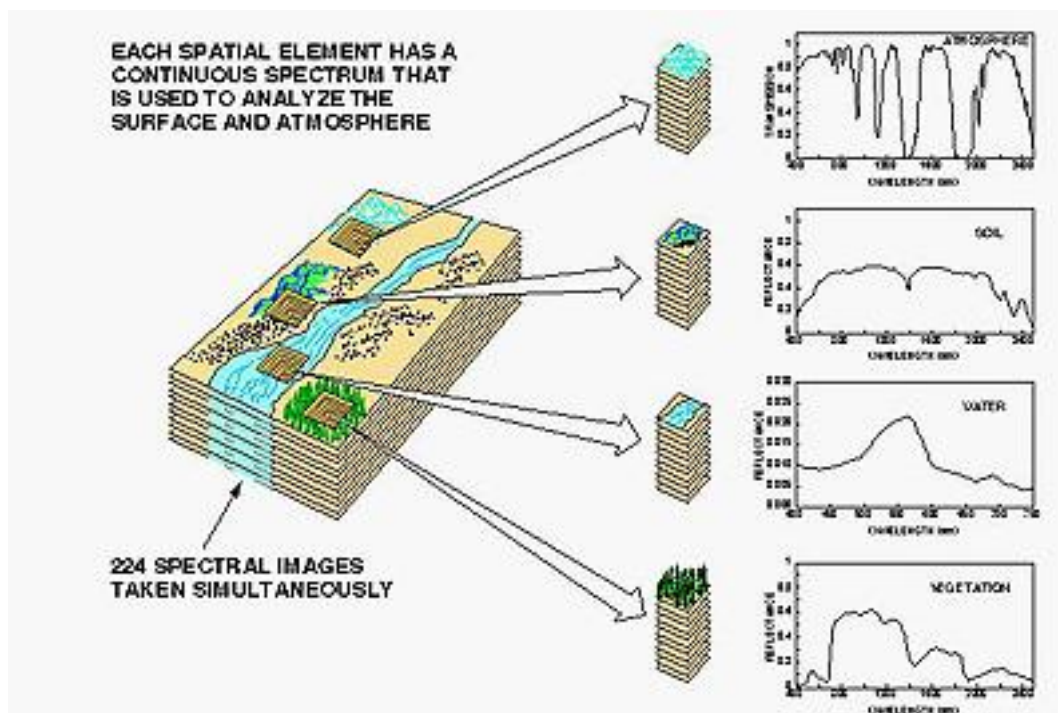


Figure 1 – Hypercube example (Short, 2009)

2.2 Interaction between the sun radiation and an object

When electromagnetic radiation hits an object several actions occur. Some of the radiation is absorbed by the object, some of it is reflected and some goes thru the object (Van der Meer, 2004). The amount of the reflected radiation is what affects the sensor. The radiation flux upholds the conservation of energy rule, as shown in [Formula 1].

$$\text{[Formula 1]} \quad \phi_{\lambda} = r_{\lambda} + \tau_{\lambda} + \alpha_{\lambda}$$

where:

r_{λ} – The flux transmitted from the object

τ_{λ} – The flux going thru the object

α_{λ} – The flux absorbed by the object

λ – Specific wavelength

An example can be shown in Figure 2, when the atmosphere is the object (an object is not necessary solid). The radiation comes from the sun (defined as a blackbody) through the atmosphere. The radiation at the top of the atmosphere is presented in yellow, and the radiation at the bottom of the atmosphere is presented in red. The red indicates the flux going thru the object (τ_{λ}), while the difference between them is the transmitted and absorbed flux ($\alpha_{\lambda} + r_{\lambda}$).

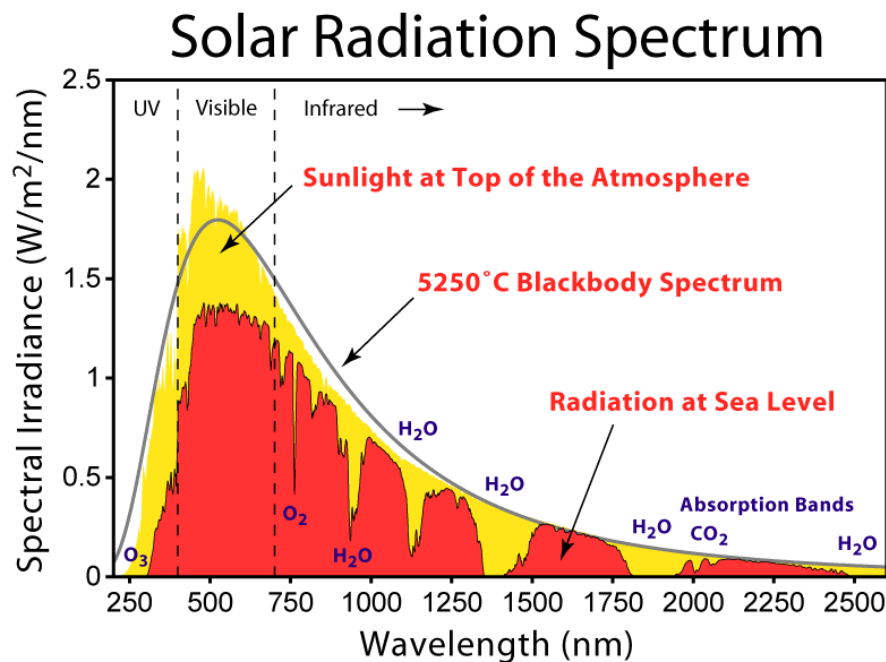


Figure 2 – Sun radiation at various places (Solar radiation handbook, 2008)

2.2.1 Absorption

Absorption radiation is the radiation that stays in the object, when radiation is transformed to another kind of energy, usually heat. Each substance is responsible for absorption at a different wavelength. In Figure 2, it is shown that at 750 nm, the atmosphere absorbed a lot of the radiation because of O₂, and at 830 nm because of H₂O.

2.2.2 Reflectance

Reflectance is defined as the ratio between the received radiation and the returned radiation, as shown in [Formula 2]. The ratio is influenced by the chemical characteristics of the object, the micro topographical surface of the object and the incident angle of the light source.

$$\text{[Formula 2]} \quad R_{\lambda} = \frac{\phi_{Out}}{\phi_{In}}$$

2.2.3 Transmittance

The transmittance is defined as the amount of energy that goes through an object. Each object transmittance is depends on the object physical and chemical characteristics.

2.3 Vegetation Spectral Behavior

Each material reflects differently the electromagnetic radiation. This section elaborates the behavior of vegetation reflectance. For vegetation, there is a known difference between the red spectrum (around 650 nm) and the NIR spectrum (around 750 nm) reflected radiation due to chlorophyll characteristics. This phenomenon is denoted as “*Red Edge*” as shown in Figure 3 and Figure 4. Green vegetation will also have a small edge around 500nm because the vegetation is green (Bradley et al., 2004).

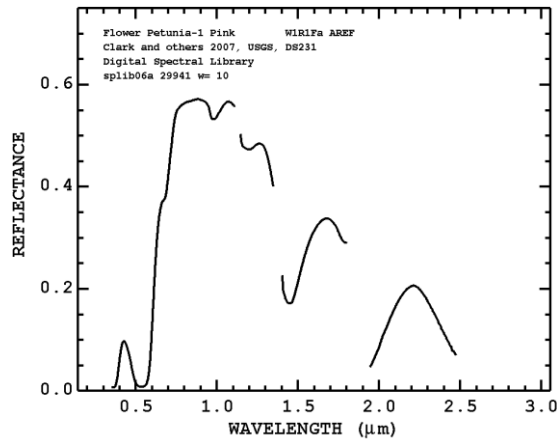


Figure 4 – Reflectance of a petunia flower
(USGS spectral library)

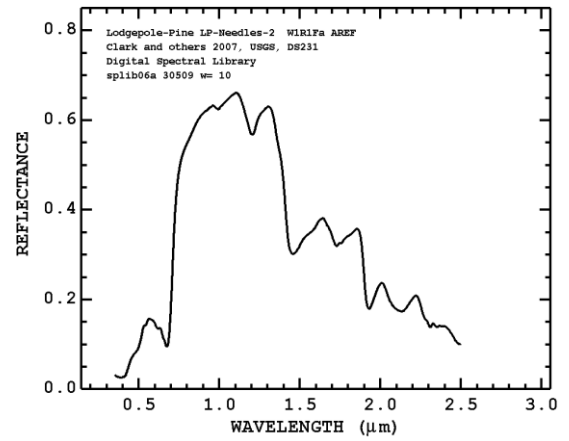


Figure 3 – Reflectance of a pine tree
(USGS spectral library)

2.4 General Hyperspectral Algorithms

In hyperspectral research, several algorithms as detailed below have been developed and tested for material detection by using the massive amount of data available.

2.4.1 PCA Transformation

The PCA transformation (Principal Component Analysis) transforms data received in a multi dimensional coordinate system to a new orthogonal coordinate system with fewer channels (Xing et al., 2007). The channels are not correlated and are called principle components. The first component has the most information about the data; the second one has the second most information and so forth. The analysis can be performed on a smaller scale of uncorrelated data (there is no obligation to select all components, only the strongest components can be taken for analysis). Reconstruction of the data from the selected components can be done by the inverse transformation, although the sample may not be reconstructed without errors due to the de-selection of all of the components. The reconstructed sample should be quite close to the original sample. Another element that affects the reconstructed sample is actually the shape of the original sample: if the original sample is smooth with little changes, the reconstructed sample should be better (there will be fewer components). The main advantage of the PCA method is its ability to deal with a small number of uncorrelated variables that allow us to do fast analysis (Xing et al., 2006). The disadvantages are the poor reconstruction and the fact that the components are mathematical numbers that do not have any relationship to the physical structure and hence, loose the spectral meaning of the material (Naganathan, 2008).

2.4.2 Channels relations

A channels relation algorithm is based on a relation between two or more channels from the reflectance radiation by a certain material (Yuan et al., 2010; Jin et al., 2010; Ye X. et al., 2008). The algorithm that is mostly used is showed in [Formula 3].

$$[\text{Formula 3}] \quad \text{Index} = \frac{C_{\text{One}} - C_{\text{Two}}}{C_{\text{One}} + C_{\text{Two}}}$$

Where:

C_{One} and C_{Two} are two specific channels that are chosen according to the desired material to be detected.

The index values go from -1 to 1; the meaning of the value changes according to the selected specific channels and material.

The advantages of the algorithm are the ability to:

- find special characteristics that directly relate for a specific material like vegetation.
- reduce the dependency of the light source intensity when one of the selected channels is very low and the other channel is very high.

The disadvantages of the algorithm are:

- it uses only a small portion of the available data.
- due to noise and changes of micro topology of the material, the channels may slightly differ for the same material which may lead to miss-classification.

2.4.3 Derivatives

The derivative method is mainly used to enhance geometry features of a spectral sample (Gong et al., 2001). When the derivative equals zero, the spectral sample has reached a local minimum or maximum and it changes its direction. Usually, for each material we can isolate several minimum and/or maximum points in a definite location, so if we encounter a derivative that equals zero in a known place, we can categorize the sample as the material. The points will not always be in the same location, but they will be in the vicinity of the expected location. Another usage of the derivative method is to discriminate between species of the same material. Different species can have a different derivative at a special wavelength. The derivative formula is shown in [Formula 4].

$$[\text{Formula 4}] \quad R'(\lambda) = \frac{dR}{d\lambda} = \frac{R(\lambda)_{i+1} - R(\lambda)_i}{\lambda_{i+1} - \lambda_i}$$

Where:

R' – The value of the derivative method.

λ_i – The wavelength in the i 'th channel.

$R(\lambda)_i$ – The intensity in the i 'th channel.

The advantages of the algorithm are the ability to:

- find special characteristics that directly relate to a specific material like vegetation or asphalt.

The disadvantages of the algorithm are:

- due to noise and changes of micro topology of the material, the location of the channels that the derivative is strong may slightly differ for the same material which may lead to misclassification.

2.4.4 Fourier analysis

Fourier analysis ([Formula 5] and [Formula 6]) allows us analyze the signal in the frequency domain, by transforming it to its cyclic components (cosine and sine, Tomiya et al., 2003). This breakdown occurs using a known FFT algorithm to transform the sample to the frequency domain. In the frequency domain, only the components with high coefficients are extracted and analyzed.

$$\text{[Formula 5]} \quad X(f) = \int_{-\infty}^{\infty} x(s)e^{-j2\pi s} ds$$

$$\text{[Formula 6]} \quad X(s) = \int_{-\infty}^{\infty} x(f)e^{-j2\pi f} df$$

Where:

$X(f)$ – The sample in the frequency domain.

$X(s)$ – The sample in the spectral domain.

The advantage of the algorithm is that a small amount of data represents almost all of the data.

The disadvantage of the algorithm is that because of the transformation to the frequency domain, the perspective of the location in the spectral domain is lost.

2.4.5 Euclidian Distance

A Euclidian distance algorithm is an algorithm that calculates the distance in the Euclidian space between the obtained sample and the average of previous recorded samples of a known material (Okamoto et al., 2006) [Formula 7].

$$[\text{Formula 7}] \quad d_E = \sqrt{\sum_{i=1}^N (x_i - u_{ki})^2}$$

Where:

N – The number of channels

x_i – The obtained sample in a specific channel

k – The number of groups that exist (different kinds of materials)

μ_{ki} – The average of a previously obtained data on a specific channel.

The advantage of the algorithm is that it uses all of the information.

The disadvantage of the algorithm is that it is sensitive to source intensity, and the pre-recorded materials must be similar.

2.4.6 Mahalanobis Distance

A Mahalanobis distance algorithm [Formula 8] is an improvement of the Euclidian algorithm by the fact that it takes into account the correlations of the dataset (DeVries, 2003).

$$[\text{Formula 8}] \quad d_M = \sqrt{(x - \mu_k)^T S^{-1} (x - \mu_k)}$$

Where:

x – The obtained vector sample

k – The number of groups that exist (different kinds of materials)

μ_k – The vector from the dataset of a specific material.

S – Covariance matrix of the dataset.

If the channels are not correlated (the covariance matrix is diagonal), the calculation will be as seen in [Formula 9].

$$[\text{Formula 9}] \quad d_M = \sqrt{\sum_{i=1}^N \frac{(x_i - u_{ki})^2}{\sigma^2}}$$

Off course, if the variance is 1, we shall receive the Euclidian distance.

The advantages of the algorithm are:

- it uses all of the information
- it uses statistical criteria to get a more precise matching to the dataset.

The disadvantage of the algorithm is that it is sensitive to source intensity.

2.4.7 Spectral Angle Mapper (SAM)

Spectral Angle Mapper is an algorithm that calculates the angle between two vectors (each vector is essentially all the wavelengths), one is a known vegetation the other is the checked pixel (Park et al., 2007; Fogler 2003). This calculation is immune to changes in the reflectance from an object caused by changes in light source intensity and incited angle. The decision if the two materials (the sample and the one comparing from the dataset) are the same is directly connected to the angle: the closer the angle to zero, the materials are more similar. The angle calculation is shown in [Formula 10].

$$\text{[Formula 10]} \quad \theta_k = \cos^{-1} \left[\frac{\sum_{i=1}^N x_i \mu_{ik}}{\sqrt{\sum_{i=1}^N x_i^2 \sum_{i=1}^N \mu_{ik}^2}} \right] = \frac{\vec{x}^T \vec{\mu}_k}{\|\vec{x}\| \|\vec{\mu}_k\|}$$

Where:

N – The number of channels

x_i – The obtained sample in a specific channel

k – The number of groups that exist (different kinds of materials)

μ_{ki} – Average of a previously obtained data on a specific channel.

The advantages of the algorithm are:

- It uses all of the information.
- It's immune to changes in intensity due to incited angle or source change.

The disadvantage of the algorithm is its sensitivity to noise, it is not equal in all the channels.

2.4.8 Summary of the algorithms

The algorithms are compared in Table 1 with applications described in Table 2.

Table 1 – Summary of the advantages and disadvantages of the algorithms

| Algorithms | Principles | Advantages | Disadvantages |
|-----------------------------|---|--|---|
| PCA Transformation | Transforms the problem dimension into a smaller one, by keeping most of the data | <ul style="list-style-type: none"> • Problem dimension is reduced | <ul style="list-style-type: none"> • Loses the perspective of the spectral domain |
| Channel Relation | Compares between two wavelengths, and check for a known difference | <ul style="list-style-type: none"> • Quick - uses two wavelengths • Immune to light changes | <ul style="list-style-type: none"> • Uses only small portion of the data |
| Derivative | Checks the derivative between several wavelengths | <ul style="list-style-type: none"> • Quick - uses several wavelengths | <ul style="list-style-type: none"> • Uses only small portion of the data |
| Fourier Analysis | Transforms the problem to the frequency domain. | <ul style="list-style-type: none"> • Problem dimension is reduced | <ul style="list-style-type: none"> • Loses the perspective of the spectral domain |
| Euclidian Distance | Compares the Euclidian distance of the sample to a known Euclidian distance of vegetation | <ul style="list-style-type: none"> • Uses all information | <ul style="list-style-type: none"> • Sensitive to light changes • The pre-recorded vegetation types must be similar |
| Mahalanobis Distance | Comparing the Mahalanobis distance of the sample to a known Mahalanobis distance of vegetation | <ul style="list-style-type: none"> • Uses all information • Uses statistical criteria for more accurate classification | <ul style="list-style-type: none"> • Sensitive to light changes |
| SAM | Considers the wavelengths as an “N” dimensional vector, and calculating the angle between the sample’s and the vegetation vector. | <ul style="list-style-type: none"> • Uses all information • Immune to light changes | <ul style="list-style-type: none"> • Relatively slow |

Table 2 – Examples of algorithms in applications

| Algorithms | Types of materials classified | Environmental Conditions | Sensors | Results |
|---------------------------------|---|---|---------------------------------|--|
| PCA Transformation | Detect bruises on apples | Indoor, constant light conditions | Hyperspectral | 97.5% |
| | Beef tenderness prediction | Indoor, constant light conditions | Hyperspectral | 96.4% |
| Channel Relation | Characterizing vegetation spectral features | Outdoor | Hyperspectral | Qualitative - showed graph |
| | Classifying vegetation, water, rock, minerals | Indoor | Multispectral | Qualitative – showed histograms |
| | Classifying ground, vegetation and obstacles | Outdoor | Multispectral and laser scanner | 85.7% for 3 classes 90.6% for vegetation classification |
| Derivative | Classify conifer species | Outdoor | Hyperspectral | 85.3% |
| Fourier Analysis | Classifying pines | Outdoor – airborne acquisition, testing and training data from the same recording | Hyperspectral | ~93% |
| Euclidian Distance | Plant classification | Outdoor | Hyperspectral | 60% |
| Mahalanobis Distance | Mapping rainforest sub-formations | Outdoor - satellite | Hyperspectral | Qualitative – showed image |
| SAM | Two plant species | Indoor | Hyperspectral | Qualitative - graph |
| | Contaminant classification | Indoor | Hyperspectral | 90% |

Chapter Three: Methodology

3.1 General

A hyperspectral data acquisition system was used to classify between vegetation and non-vegetation in an outdoor environment. Classification algorithms were developed and adapted.

3.1.1 Assumptions

This study assumes the following:

- Off-line classification is conducted (classification is not in real-time due to sensor limitations).
- During recording of a full dataset (i.e., when sampling all of the wavelengths along approximately 30 seconds), there are no changes in the environment.
- There is no knowledge about the external light conditions (like direct sunlight, clouds and shadows, indirect light).

3.2 Experimental Setup

The hyperspectral system was composed of commercial components as described in Figure 5 and Figure 6. The spectral component is selected using an acousto-optic tunable filter (AOTF) which acts as an electronically tuned bandpass filter. The available wavelengths are 500-900 nm with a 5nm FWHM. The image sampled by the AOTF filter is captured by a black and white CCD cooled camera (COOL-1300Q/QC by VDS Company) with a pixel resolution of 1280X1024, and 640X512 with 2x2 binning technology. Binning technology is a special technology in which each 4 adjunct pixels serve as 1 pixel, resulting in increased light for each pixel. The lens angle was 12° horizontally and 9° vertically. The control of the AOTF was done by a Direct Digital Synthesizer (DDS) which sends a RF wave to the AOTF (through an amplifier), and thereby changes the filter characteristics.

Seven hypercubes¹ were taken in different scenarios, at different hours of the day, in order to test different light conditions with different types of vegetation (Figure 7). Each hypercube had at least 30% vegetation in the image.

¹ A hypercube is a three dimensional cube which has two spatial dimensions (x,y) and one spectral dimension (z). The value of each pixel in the two spatial dimensions is the returned electromagnetic radiation at that pixel. The z axis includes all the wavelengths images (i.e., one image for each wavelength).

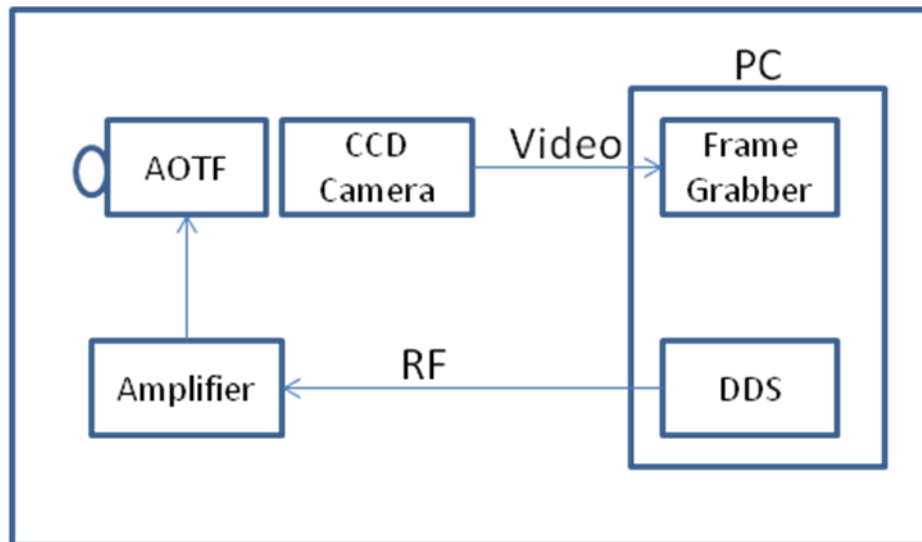


Figure 5 – Hyperspectral imaging system architecture

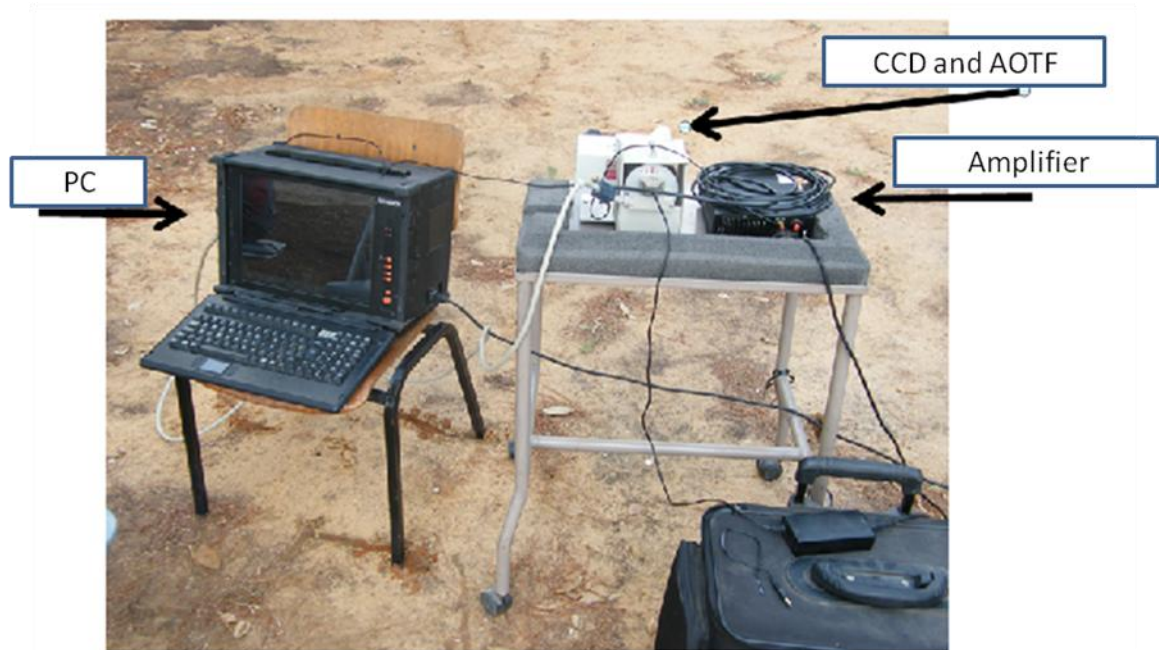


Figure 6 – Photograph of the hyperspectral imaging system



Figure 7 – Different types of environments

3.3 Algorithms

The following algorithms were developed and adapted to classify between vegetation and non vegetation:

1. Mahalanobis distance
2. Derivative
3. Spectral Angle Mapper (SAM)
4. Minor Spectral Angle Mapper (MSAM)
5. Channel relation (NDVI)
6. Channel relations Tree (NDVI_T)
7. Dynamic Channel relations

Specific new developments in each algorithm are detailed in Chapter Four.

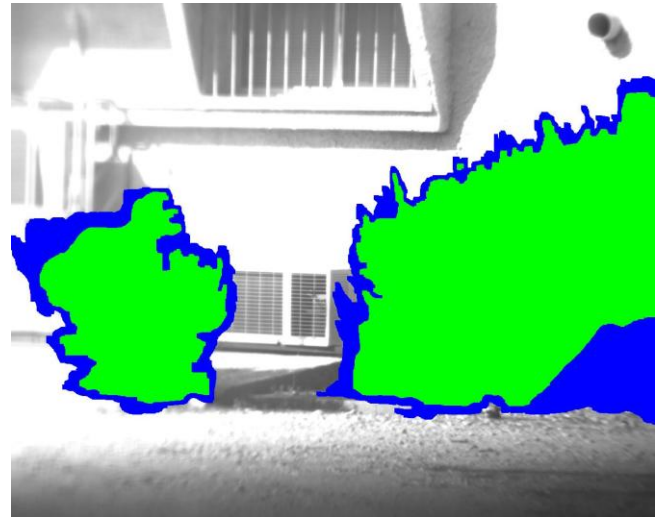
3.4 Performance Measures

Three performance measures were used for evaluation (Table 3): Total Success Rate (number of correct detections), False Positive Rate and False Negative Rate. The measures were calculated from the confusion matrix (Table 3) by comparing the result of the algorithms (the out coming classified image) to a manual classified image and are detailed below.

The manual classified image was created by taking the captured images at a specific wavelength (590nm), manually marking in green the pixels which were vegetation and in blue the pixels that were on the edges, and as such could be either vegetation or non-vegetation. All other pixels were considerate non-vegetation. See Figure 8 for an example. All manually marked figures are detailed in Appendix E.



a – image at 590nm



b – manually classified image

Figure 8 – Example of an image and a manual classified image

Table 3 – Confusion Matrix

| | | Actual state | |
|-------------------|----------------|----------------------------------|----------------------------------|
| | | Vegetation | Non-vegetation |
| Classifier Result | Vegetation | True Positive | False positive (Type 1 Error) |
| | Non-vegetation | False Negative (Type 2 Error) | True Negative |

Total Success Rate

The total success rate is calculated according to [Formula 11]. This measure states the number of correctly classified pixels relative to all the total number of pixels in the image.

[Formula 11]
$$total\ success\ rate = \frac{(True\ Positive + True\ Negative)}{All\ of\ the\ pixels}$$

False Positive Rate

The False Positive rate is calculated according to [Formula 12]. This measure states how many non-vegetation pixels were classified as vegetation from all of the non-vegetation pixels.

$$\text{[Formula 12]} \quad \textit{false positive rate}(\alpha) = \frac{\textit{False Positive}}{(\textit{False Positive} + \textit{True Negative})}$$

False Negative Rate

False Negative Rate is calculated according to [Formula 12]. This measure states how many vegetation pixels were classified as non-vegetation from all of the vegetation pixels.

$$\text{[Formula 13]} \quad \textit{false negative rate}(\beta) = \frac{\textit{False Negative}}{(\textit{False Negative} + \textit{True Positive})}$$

3.5 Decision tree

The decision tree is one of the most widely used and practical methods for inference and classification. A decision tree can be represented as sets of “if-then” rules. In order to choose which of the attribute (inputs) best discriminates the training data, usually a statistical property known as information gain is used. Information gain is the expected reduction in entropy (uncertainty) caused by partitioning the examples (the training data) according to an attribute. This information gain method does not assume any statistical properties on the data itself (e.g., normal distribution) and such is best suited in this case, when the statistical distribution is unknown.

When building a decision tree, the over-fitting phenomenon may arise. Over-fitting is, modeling the training data too well, and as a result, representing a particular case and not the general one. Over-fitting is represented in the decision tree as a deep tree with many levels. To avoid over-fitting, a simple method was applied: assuming a minimal number of instances (in our case - pixels) in each leaf, thus not allowing the tree to grow more and more.

3.6 Calibration

In sensing, usually there is a need to calibrate each sensor. This is commonly done by using a white paper, which serves as a reference. However, this requires accurate procedures such as using the same type of white paper, and applying the same illumination source located in the same position (range and orientation). Since it is usually very problematic to ensure identical operating conditions (e.g. even if the light source is from the same type, if it was working for a long time, its behavior is different) this thesis employed a different approach for calibration. Instead of using the same algorithm and its parameters, and bringing the new sensor to the old sensor performances, the opposite approach was chosen. For every sensor, recordings were made, and new parameters were chosen for each specific algorithm (as described in Chapter Four) to ensure it fits the "old" sensor. While this procedure requires more time to adjust for a new sensor, it ensures optimal fit to that sensor.

3.7 Clouds interference

To check how the clouds affect the transmission of the light coming from the sun, several recordings were conducted on a white paper for all the wavelengths. Several recordings were done when the sun was without any interference, and the light hit the white paper directly. Other recordings were made when there were lots of clouds that blocked the sun completely. To neglect the exposure time, which is different for each recording, especially between the direct sun and the clouds, all data was normalized. The normalization for each recording was made by subtracting the minimum value from each wavelength, and dividing each wavelength by the maximum value, resulting in values between zero and one [Formula 14]. Ten different pixels in different areas in the image were taken and the mean was calculated for every wavelength. This was repeated for all of the recordings (direct sunlight and clouds).

[Formula 14]
$$Wave_i = \frac{Wave_i - \min(Waves)}{\max(Waves) - \min(Waves)}$$

Where:

Wave_i – The wavelength to be normalized.

Waves – all of the wavelengths.

3.8 Sensitivity Analysis

Several sensitivity analyses were conducted for each algorithm as detailed below.

3.8.1 Independent evaluation of hypercubes

For each algorithm, each hypercube was tested independently: half of the pixels from each hypercube were chosen randomly for the training data and the rest were used as the testing data.

Each hypercube behaves differently due to the different conditions in the environment (e.g., light).

3.8.2 Cross Validation

Each of the algorithms was tested on all of the hypercubes. The training data consisted of half of the hypercubes chosen randomly while the testing data was the remaining data.

3.8.3 Training by different hypercubes

The training data consisted of two hypercubes chosen randomly, while the testing data were the five remaining hypercubes. This was done twice, each time with different hypercubes serving as training data.

3.8.4 Light conditions

To check the algorithm's robustness to variance in light conditions, each hypercube was checked under different light conditions. The light conditions were manually categorized into three sections: Strong, Medium and Low. For three of the hypercubes the medium and low categories were created by multiplying the hypercubes by a factor of 0.9 and 0.7 accordingly. The other two hypercubes were taken with smaller exposure times representing low lighting conditions.

3.8.5 Noise

In order to check the robustness to noise, noise was added to each wavelength, for every pixel in the testing data. The level of the added noise varied from 1% to 10%. The testing data consist of five hypercubes randomly selected.

3.8.6 The number of hypercubes for training

In order to check the susceptibility of the algorithm to the number of training hypercubes, additional training sets with four and five hypercubes as training sets were evaluated.

Chapter Four: Algorithms

4.1 Overview

Seven algorithms were developed and adapted for classifying between vegetation and non vegetation: Mahalanobis distance, Derivative, SAM, MSAM, NDVI, NDVI_T and dynamic channel relation.

The specific developments included:

- *Optimal parameters were found* for each algorithm by developing decision trees.
- *The specific derivatives for the Derivative algorithm were chosen automatically* by a decision tree.
- The *MSAM* algorithm is a variation of the *SAM* algorithm; this is a **new algorithm developed in this research** in order to check the assumption that most important information is located in only a few wavelengths. *The important wavelengths were chosen* automatically by a decision tree in this algorithm and in the *NDVI* algorithm.
- The *NDVI_T* algorithm is a variation of the *NDVI* algorithm, is a **new algorithm developed in this research** in order to check if multiple *NDVI* algorithms can improve performance.
- The *Dynamic Channel Relation* is another variation of the *NDVI* algorithm, also **developed in this research**, in order to check the behavior of the *NDVI* algorithm, using different exposure time settings for the two wavelengths.

4.2 Mahalanobis distance

The *Mahalanobis distance* algorithm requires a reference of the vegetation. To implement the algorithm, we assumed that none of the channels are correlated, thus enabling us to use [Formula 15].

To determine the vegetation reference, several steps were conducted: first, the mean and the standard deviation of all vegetation pixels were calculated from the training data. Second, the *Mahalanobis distance* was calculated for every pixel in the training data (vegetation and non-vegetation alike). In order to optimally decide on the correct *Mahalanobis distance* representing vegetation, a “C5.0” decision tree algorithm was used.

$$\text{[Formula 15]} \quad d_M = \sqrt{\sum_{i=1}^N \frac{(x_i - u_{ki})^2}{\sigma^2}}$$

Where:

x_i – The obtained vector sample.

k – The number of groups that exist (different kinds of materials under different lighting conditions).

μ_{ki} – The vector from the dataset of a specific material.

σ – Standard deviation of a specific channel.

Both vegetation and non-vegetation distances were entered into the decision tree algorithm as inputs; the output was the actual tree design. To avoid over-fitting, the decision tree was designed so that each of the leaves of the decision tree contained at least 1/3 of the amount of all the pixels, thus creating a decision tree with 1 root and two leaves. The optimal distance for deciding if a pixel is vegetation or non-vegetation is equivalent to the condition to go to the “vegetation” leaf from the root. For the testing set, the *Mahalanobis distance* of each pixel was calculated and compared to the optimal vegetation distance derived from the decision tree. A pixel with a distance greater than the *Mahalanobis distance* was classified as non-vegetation; a pixel with a distance smaller or equal to the *Mahalanobis distance* was classified as vegetation.

4.3 Derivative algorithm

The *derivative* algorithm requires calculation of all possible adjacent derivatives. Since there are 81 possible wavelengths, 80 derivatives were calculated between each wavelength and its adjunct wavelength. To optimally select the most important derivatives, the “C5.0” decision tree algorithm was used. As inputs, all of the derivatives of every pixel from the training set were entered; the output was the actual tree design. To avoid over-fitting, the decision tree was designed so that each of the leaves of the decision tree contained at least 10% of the amount of all the pixels, thus creating a decision tree not too short (so it can consider several derivatives), but still without the over-fitting problem. The decision tree that was created is depicted in Figure 9. For the testing set, we calculated only the derivatives that appear in the decision tree, and just followed the tree from the root to the bottom leaf.

The wavelengths received by the vegetation were between 650 to 735 nm. The fact that the most important derivative are in this area, also known as the “Red Edge” for vegetation, reinforces the correctness of the implementation.

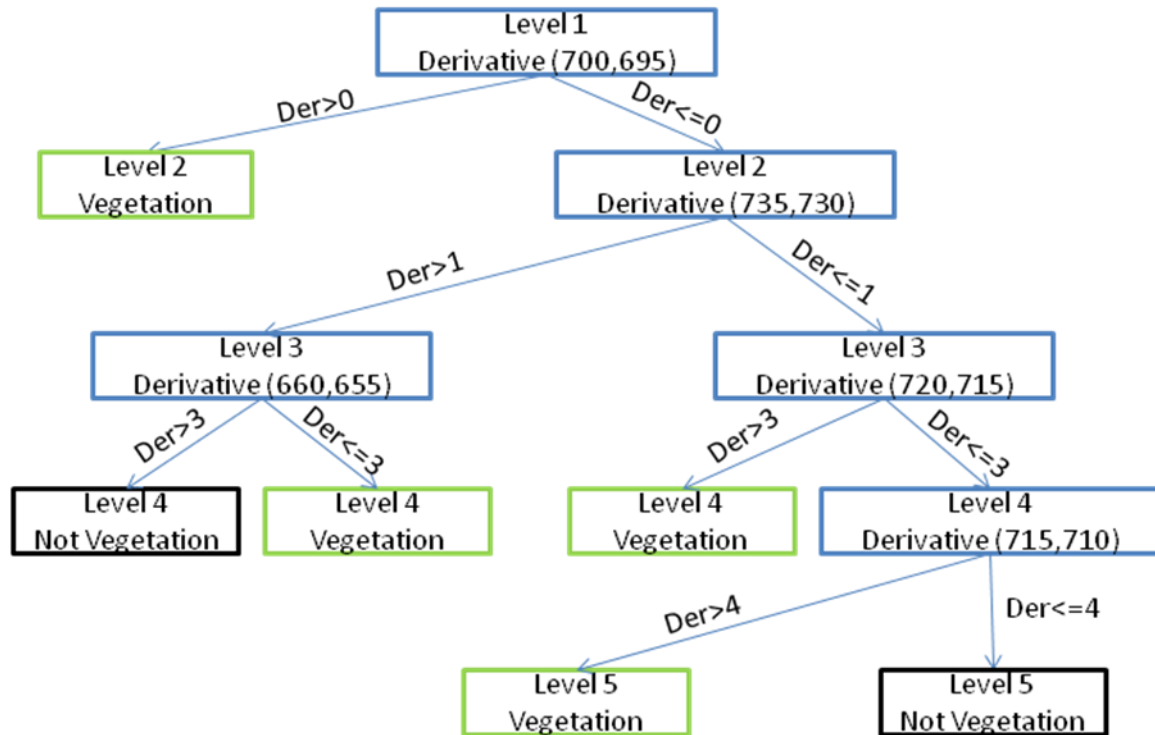


Figure 9 – Derivative, vegetation decision tree

4.4 Spectral Angle Mapper (SAM)

The *SAM* algorithm requires references from the vegetation to be classified. The vegetation reference was created by calculating the average of every wavelength, when the data is all of the vegetation pixels in the training set. All of the wavelengths were taken into consideration. Then, we calculated the angle between each pixel in the training to the reference according to [Formula 16]. To optimally choose the threshold that corresponds to vegetation, the “C5.0” decision tree algorithm was used. Both vegetation and non-vegetation angles were entered into the decision tree algorithm as inputs; the output was the actual tree design. To avoid over-fitting, the decision tree was designed so that each of the leaves of the decision tree contained at least 1/3 of the amount of all the pixels, thus creating a decision tree with one root and two leaves. The optimal angle for deciding if a pixel is vegetation or non-vegetation is equivalent to the condition to go to the “vegetation” leaf from the root. For the testing set, the *angle* of each pixel was calculated (by comparing it to the reference vegetation) and compared to the optimal vegetation angle derived from the decision tree. A pixel with an angle greater than the *optimal angle* was classified as non-vegetation; a pixel with an angle smaller or equal to the optimal angle was classified as vegetation.

$$[Formula\ 16] \quad \theta_k = \cos^{-1} \left[\frac{\sum_{i=1}^N x_i \mu_{ik}}{\sqrt{\sum_{i=1}^N x_i^2 \sum_{i=1}^N \mu_{ik}^2}} \right] = \frac{\vec{x}^T \vec{\mu}_k}{\|\vec{x}\| \|\vec{\mu}_k\|}$$

Where:

N – The number of channels.

x_i – The obtained sample in a specific channel.

k – The number of groups that exist (different kinds of materials).

μ_{ki} – Average of a previously obtained data on a specific channel.

4.5 Minor Spectral Angle Mapper (MSAM)

The Minor Spectral Angle Mapper algorithm process is done exactly like the *SAM* algorithm except for one change: instead of using all of the 81 wavelengths, the algorithm uses only the most five important wavelengths. To select the most effecting wavelengths, the “C5.0” decision tree algorithm was used. The inputs to the decision tree were the entire 81 wavelengths; the output was the classification tree (Appendix C). The upper five wavelengths (the wavelengths that are closer to the root) were chosen, as they are the most influencing ones. This method of optimally choosing the most important wavelengths and using only them was specially developed in this thesis in order to obtain performance with a low number of wavelengths.

4.6 Channel relation (NDVI)

The channel relation algorithm requires selection of three parameters from the channel relation equation [Formula 3]: Channel1, Channel 2 and “Index”. The acquisition system provides 81 channels, so there are 3240 options [Formula 17] for checking all of the possibilities of the “Channel1” and “Channel2”. In addition, it is necessary to determine the “Index” parameter.

$$[\text{Formula 17}] \quad \sum_{i=1}^{80} i = \frac{80 \cdot 81}{2} = 3240$$

To optimally select these three parameters, the “C5.0” decision tree algorithm was used. To select the optimal channels and the threshold, two steps were conducted. First, the entire 81 channels were entered as inputs to the decision tree. Then, the most important wavelengths were taken from the decision tree by taking the upper nodes of the tree (in a decision tree – the higher the node, the wavelength is more important, Appendix C). After receiving only five wavelengths, all of the possible “channel relations” options were calculated and re-entered to the “C5.0” algorithm by creating a new decision tree. The root of the new tree is the final “channel relation” and it also contains the required “Index” parameter (Figure 10).

The wavelengths for vegetation were accordingly selected between 665 and 800 nm. The fact that the most important channel relation is in the area of the known “Red Edge” for vegetation (Fogler 2003), reinforces the correctness of the implementation part. The channel relation for vegetation is also called *NDVI* (Normalized Difference Vegetation Index).

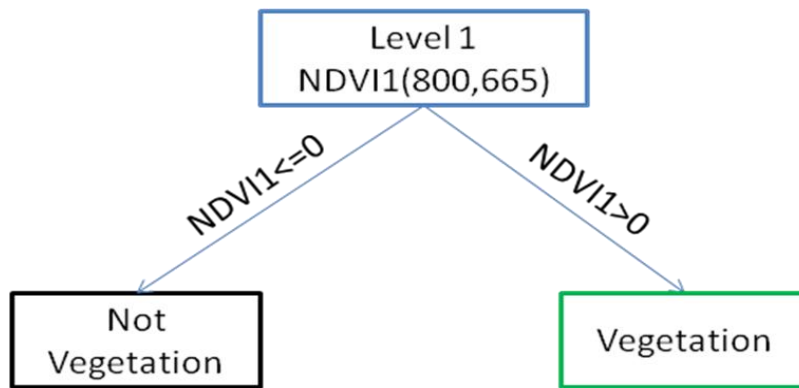


Figure 10 –Channel relation - vegetation decision tree

4.7 Channel relations tree (NDVI_T)

This algorithm is an extension of the single channel algorithm which was **specially developed in this thesis**. It is calculated exactly the same as the previously *channel relation*, except for one change: instead of taking only the root in the second decision tree, the entire tree was considered. This will use several *channel relations*, each with different wavelengths.

The final decision tree is depicted in Figure 11:

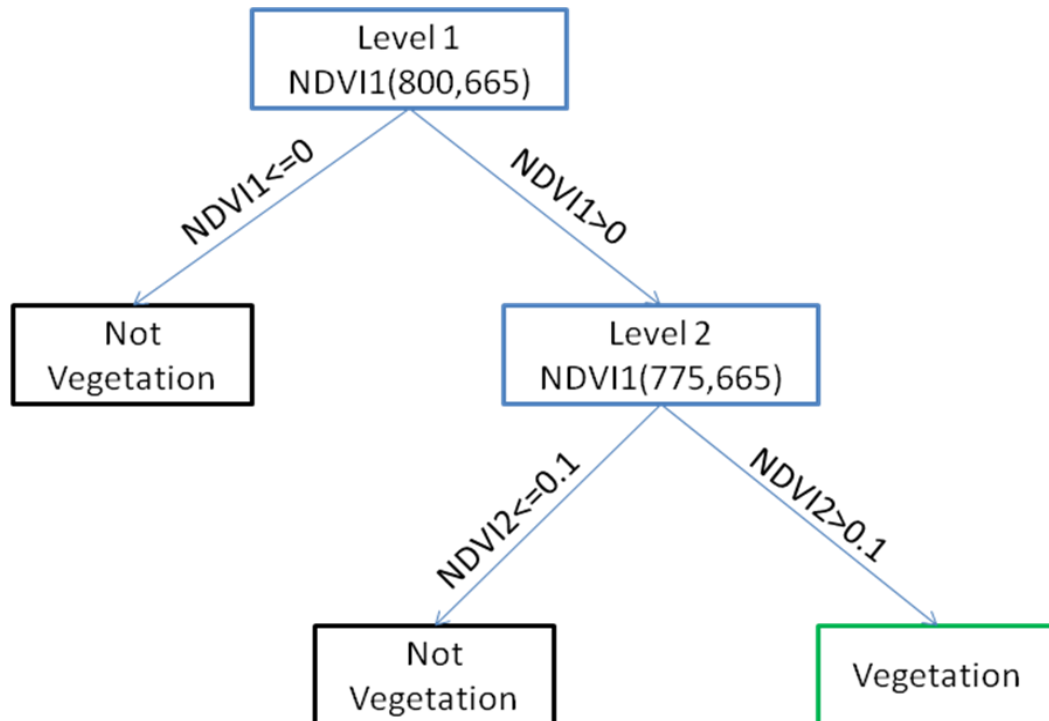


Figure 11 – Multiple channel relation - vegetation decision tree

4.8 Dynamic Channel relations

The dynamic channel relation is a variation of the “regular” channel relation algorithm **specially developed in this thesis** in order to check the effect of exposure time for receiving better results. The resulting intensity for each wavelength is influenced by several factors such as light source intensity, atmosphere, filter characteristics, CCD characteristics and exposure time. These factors behave differently from wavelength to wavelength. Most of the factors cannot be controlled, but the exposure time can be controlled. Giving a longer exposure time to a specific wavelength is equal to multiplying its current intensity with a weight. As opposed to the previous *NDVI* which uses [Formula 3], the dynamic channel relation uses a different weight for each wavelength, as seen in [Formula 18]. This was implemented in simulation. From the hypercubes of the training set only the two wavelengths that were found in *NDVI* were taken. In each simulation, each wavelength was multiplied by its weight. The weights were changed between 0.2 - 2, in 0.2 intervals. If the value was greater than the maximum (implying saturation condition), it was changed to be the maximum. There were a total of 200 simulations to cover all possibilities. For each simulation, the ‘Index’ was calculated exactly like the *NDVI* – creating a decision tree using a C5.0 algorithm, and extracting the Index from the first level of the decision tree.

$$[\text{Formula 18}] \quad \text{Index} = \frac{\alpha C_{\text{One}} - \beta C_{\text{Two}}}{\alpha C_{\text{One}} + \beta C_{\text{Two}}}$$

Where:

C_{One} and C_{Two} are two specific channels that are chosen according to the desired material to be detected.

α is the weight for C_{One} .

β is the weight for C_{Two} .

Index is the criteria to choose if the material is vegetation or not.

Chapter Five: Results and Discussion

5.1 Mahalanobis distance

5.1.1 Independent evaluation of hypercubes

Result indicated that the algorithm is not good for vegetation classification (Table 4): The average success rate was 85.7%, the FPR (False Positive rate) was 12.3% and the FNR (False Negative Rate) was 14.3%. All three measures had high standard deviations (approximately 10%) implying inconsistent results which are highly dependent on the examined hypercube.

Table 4 – Mahalanobis - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Hypercube1 | 0.836 | 0.066 | 0.208 |
| Hypercube2 | 0.913 | 0.084 | 0.090 |
| Hypercube3 | 0.758 | 0.111 | 0.293 |
| Hypercube4 | 0.955 | 0.047 | 0.044 |
| Hypercube5 | 0.935 | 0.012 | 0.097 |
| Hypercube6 | 0.679 | 0.449 | 0.202 |
| Hypercube7 | 0.925 | 0.094 | 0.064 |
| Average | 0.857 | 0.123 | 0.143 |
| Std | 0.104 | 0.147 | 0.092 |

5.1.2 Cross Validation

The cross validation results (Table 5) indicates low classifications results (only 85% total success with approximately 14.5% FPR and FNR). This implies that the algorithm has a problem with training data consisting of various illumination conditions.

Table 5 – Mahalanobis – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|---------------------------|----------------------------|----------------------------|
| 0.857 | 0.152 | 0.140 |

5.1.3 Training by two hypercubes

Two hypercubes were used as the training data and the other five hypercubes were the testing data (Table 6). The difference in the total success for the two training sets, suggests that the algorithm is very sensitive

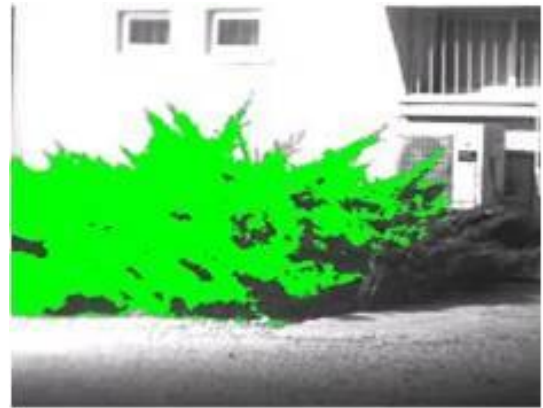
to the training set. Even in the better training set, best performance yielded only 86% success, with a FNR of 10.6% and a very high FPR (42%) implying poor classification.

Table 6 – Mahalanobis – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Training Set 1 | 0.860 | 0.421 | 0.106 |
| Training Set 2 | 0.543 | 0.752 | 0.008 |
| Average | 0.701 | 0.586 | 0.057 |
| Std | 0.224 | 0.234 | 0.070 |



a – real image



b – classified image

Figure 12 – Example of Mahalanobis distance classification

5.1.4 Different light conditions

Results (Table 7) indicate that the *Mahalanobis distance* algorithm behaves the same under different light conditions, but still performs poorly with average success of 87.4% and a FPR of 38.9% and FNR of 10.8%.

Table 7 – Mahalanobis – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Strong | 0.860 | 0.421 | 0.106 |
| Medium | 0.880 | 0.284 | 0.105 |
| Low | 0.881 | 0.462 | 0.112 |
| Average | 0.874 | 0.389 | 0.108 |
| Std | 0.012 | 0.093 | 0.004 |

5.1.5 Noise

Results indicate that noise up to 10% does not affect the algorithm's performances (Table 8). There is slight increase in FNR and decrease in FPR for 10% noise as compared to 8% noise which has no effect.

Table 8 – Mahalanobis – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------------|----------------------------|----------------------------|
| 1% | 0.859 | 0.417 | 0.112 |
| 2% | 0.860 | 0.415 | 0.113 |
| 5% | 0.863 | 0.404 | 0.120 |
| 7% | 0.869 | 0.386 | 0.126 |
| 8% | 0.869 | 0.360 | 0.141 |
| 10% | 0.853 | 0.275 | 0.167 |

5.1.6 Training by different number of hypercubes

Results indicate a drastic decrease in performance with an average of 56.9% when using four and five hypercubes as training sets. The FPR and FNR are also higher with a 55.8% and 20.9% respectfully. This result is expected due to the fact that the *Mahalanobis distance* algorithm calculates the normalized distances in the variance, and when there are lots of vegetation pixels taken under different light conditions, the variance for each wavelength is big, and the difference between the distance that represent vegetation and the distance that represent non-vegetation is very small.

Table 9 – Mahalanobis – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| 4 Hypercubes | 0.630 | 0.488 | 0.166 |
| 5 Hypercubes | 0.508 | 0.558 | 0.209 |
| Average | 0.569 | 0.523 | 0.187 |
| Std | 0.086 | 0.049 | 0.031 |

5.2 Derivative

5.2.1 Independent evaluation of hypercubes

Almost all of the hypercubes received more than 94% total success (Table 10) except for one hypercube which resulted in a low total success (Hypercube6). The average success rate was 94.9%, the FPR was 7% and the FNR was 3.7%. The high success rate implies that the algorithm is good for classifying under similar light conditions. All three measures had low standard deviations (approximately 3%) which implies consistency of the algorithm.

Table 10 – Derivative - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Hypercube1 | 0.948 | 0.065 | 0.041 |
| Hypercube2 | 0.945 | 0.064 | 0.047 |
| Hypercube3 | 0.958 | 0.046 | 0.039 |
| Hypercube4 | 0.974 | 0.036 | 0.019 |
| Hypercube5 | 0.971 | 0.050 | 0.012 |
| Hypercube6 | 0.886 | 0.173 | 0.077 |
| Hypercube7 | 0.962 | 0.059 | 0.026 |
| Average | 0.949 | 0.070 | 0.037 |
| Std | 0.030 | 0.046 | 0.022 |

5.2.2 Cross Validation

The cross validation results (Table 11) indicates very good classifications results (total success of 94% with FPR of 12.6% and FNR of 2.9%). This implies that the algorithm can classify well, when the training and testing data are with the same lighting conditions.

Table 11 – Derivative – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|---------------------------|----------------------------|----------------------------|
| 0.941 | 0.126 | 0.029 |

5.2.3 Training by two hypercubes

Two hypercubes were used as the training data and the other five hypercubes were the testing data (Table 12). In this case the algorithm tends to have a problem with classifying correctly other vegetations in different light conditions with a total success of 86% (for both training sets). The FPR is very high in this case, because pixels were classified as vegetation when there was a change from shadow to light. In both training sets, the FPR and FNR are approximately the same (with low standard deviations).

Table 12 – Derivative – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| Training Set 1 | 0.863 | 0.494 | 0.017 |
| Training Set 2 | 0.879 | 0.427 | 0.047 |
| Average | 0.871 | 0.460 | 0.032 |
| Std | 0.012 | 0.047 | 0.021 |



a – real image



b – classified image

Figure 13 – Example of Derivative classification

5.2.4 Different light conditions

The results (Table 13) show that the *Derivative* algorithm maintained medium performances, when the light conditions changes with average success of 87.3% and a FPR of 46.6% and FNR of 1.6%.

Table 13 – Derivative – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| Strong | 0.863 | 0.494 | 0.017 |
| Medium | 0.867 | 0.481 | 0.015 |
| Low | 0.889 | 0.424 | 0.016 |
| Average | 0.873 | 0.466 | 0.016 |
| Std | 0.014 | 0.037 | 0.001 |

5.2.5 Noise

Results indicate that noise affects the performances even with low noise of 1% (Table 14). The total success is lower (83.2 vs. 87.3), the FPR is higher (53.7 vs. 46.6) and the FNR is slightly higher (2 vs. 1.6).

Table 14 – Derivative – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|-------------|--------------------|---------------------|---------------------|
| 1% | 0.832 | 0.537 | 0.020 |
| 2% | 0.800 | 0.577 | 0.030 |
| 5% | 0.764 | 0.611 | 0.056 |
| 7% | 0.756 | 0.617 | 0.078 |
| 8% | 0.756 | 0.617 | 0.080 |
| 10% | 0.757 | 0.616 | 0.080 |

5.2.6 Training by different number of hypercubes

It shows a drastic increase in performances when using four hypercubes as training sets (92% total success, 15% FPR and 1.2% FNR), however, performance decreases when using five hypercubes. The reason for this is that the hypercubes are taken in different light conditions, and hence, behaves differently.

Table 15 – Derivative – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| 4 Hypercubes | 0.920 | 0.152 | 0.012 |
| 5 Hypercubes | 0.825 | 0.135 | 0.183 |
| Average | 0.872 | 0.144 | 0.098 |
| Std | 0.067 | 0.011 | 0.121 |

5.3 Spectral Angle Mapper (SAM)

5.3.1 Independent evaluation of hypercubes

The *SAM* algorithm was tested on each hypercube on its own (Table 16). Almost all of the hypercubes received more than 92% total success except for one hypercube which resulted in a low total success of only 80% total success and 23% of FNR (Hypercube6). The average success rate was 93.9%, with a FPR of 4.8% and of FNR 5.8%. The high success rate implies the algorithm is good for vegetation classification under the same lighting conditions. The high standard deviation (8.1%) of the FNR is due to hypercube6, and without it, it is much lower (3.1%).

Table 16 – SAM - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Hypercube1 | 0.949 | 0.073 | 0.033 |
| Hypercube2 | 0.970 | 0.054 | 0.005 |
| Hypercube3 | 0.957 | 0.051 | 0.036 |
| Hypercube4 | 0.986 | 0.019 | 0.010 |
| Hypercube5 | 0.985 | 0.026 | 0.006 |
| Hypercube6 | 0.804 | 0.061 | 0.230 |
| Hypercube7 | 0.924 | 0.052 | 0.087 |
| Average | 0.939 | 0.048 | 0.058 |
| Std | 0.064 | 0.019 | 0.081 |

5.3.2 Cross Validation

The cross validation results (Table 17) indicates very good classifications results (high total success) and low FPR and FNR. This implies that the algorithm can classify well, when the training and testing data are with the same lighting conditions.

Table 17 – SAM – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|---------------------------|----------------------------|----------------------------|
| 0.955 | 0.077 | 0.032 |

5.3.3 Training by two hypercubes

The high total success criteria (Table 18) shows that even when hypercubes are taken in different scenarios with different light conditions, there is a common element between all of the hypercubes. Both the FPR and the FNR are approximately the same (low standard deviation) which implies that the algorithm robustness.

Table 18 – SAM – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Training Set 1 | 0.972 | 0.071 | 0.019 |
| Training Set 2 | 0.974 | 0.047 | 0.027 |
| Average | 0.973 | 0.059 | 0.023 |
| Std | 0.001 | 0.017 | 0.005 |



a – real image



b – classified image

Figure 14 – Example of SAM classification

5.3.4 Different light conditions

The results (Table 19) indicate that the *SAM* algorithm behaves the same under different light conditions, as expected with average success of 97.2% and a FPR of 6.5% and FNR of 1.9%.

Table 19 – SAM – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Strong | 0.972 | 0.071 | 0.019 |
| Medium | 0.972 | 0.065 | 0.019 |
| Low | 0.972 | 0.060 | 0.020 |
| Average | 0.972 | 0.065 | 0.019 |
| Std | 0.000 | 0.006 | 0.000 |

5.3.5 Noise

It shows that noise until 2% does not affect performance (Table 20). There is a small decrease in performances with noises up to 8% (although the FPR is lower, the FNR is higher). For noises above 8% performance greatly decreases with higher FNR (11.7 vs. 7.6) and lower total success (90.4 vs. 93.6).

Table 20 – SAM – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------------|----------------------------|----------------------------|
| 1% | 0.972 | 0.071 | 0.020 |
| 2% | 0.972 | 0.068 | 0.021 |
| 5% | 0.965 | 0.058 | 0.034 |
| 7% | 0.947 | 0.055 | 0.062 |
| 8% | 0.936 | 0.057 | 0.076 |
| 10% | 0.904 | 0.058 | 0.117 |

5.3.6 Training by different number of hypercubes

It shows a drastic decrease in performances when using four and five hypercubes as training sets (Table 21). The total success reduces from 97% to 67%, the FPR increased to 40% from 6.5% and the FNR increased to 29.8% from 1.9%. This is expected due to the fact that the *SAM* algorithm calculates the angle between a known vegetation vector to the pixel vector, and when we are using multiple training sets that were acquired in different light conditions, the known vegetation vector (calculated by the average of the vegetation in the training set) behaves differently than real vegetation.

Table 21 – SAM – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| 4 Hypercubes | 0.674 | 0.441 | 0.257 |
| 5 Hypercubes | 0.671 | 0.361 | 0.338 |
| Average | 0.672 | 0.401 | 0.298 |
| Std | 0.002 | 0.056 | 0.057 |

5.4 Minor Spectral Angle Mapper (MSAM)

5.4.1 Independent evaluation of hypercubes

Almost all of the hypercubes received more than 91% total success except for two hypercubes which resulted in low total success of 75% and high FPRs and FNRs (10 and 38.6%; 26.8 and 13.6% respectively for Hypercube6 and Hypercube7, Table 22). The average success rate was 89.8%, with 8.7% FPR and 9.8% FNR. The high success rate means the algorithm is good for vegetation classification under the same lighting conditions in most environments. The high standard deviation in all three measures is because of hypercubes 6 and 7 results.

Table 22 – MSAM - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Hypercube1 | 0.910 | 0.035 | 0.124 |
| Hypercube2 | 0.968 | 0.047 | 0.017 |
| Hypercube3 | 0.947 | 0.015 | 0.080 |
| Hypercube4 | 0.973 | 0.012 | 0.037 |
| Hypercube5 | 0.979 | 0.015 | 0.026 |
| Hypercube6 | 0.759 | 0.102 | 0.268 |
| Hypercube7 | 0.750 | 0.386 | 0.136 |
| Average | 0.898 | 0.087 | 0.098 |
| Std | 0.101 | 0.135 | 0.088 |

5.4.2 Cross Validation

The cross validation results (Table 23) indicates very good classifications results (high total success and low FPR and FNR), although lower than the results obtained by the *SAM* algorithm. This implies that the algorithm can classify well, when the training and testing data are with the same lighting conditions.

Table 23 – MSAM – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|---------------------------|----------------------------|----------------------------|
| 0.933 | 0.132 | 0.039 |

5.4.3 Training by two hypercubes

Two hypercubes were used as the training data and the other five hypercubes were the testing data (Table 24). The high total success criteria show that most of the information that distinct the vegetation from the non-vegetation is found in the five wavelengths. The FPR is lower than the FNR in the first training set, and the opposite occurs in the second training set; this implies that performance depends on the training sets much more than the *SAM* algorithm.

Table 24 – MSAM – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Training Set 1 | 0.941 | 0.062 | 0.073 |
| Training Set 2 | 0.957 | 0.198 | 0.040 |
| Average | 0.949 | 0.130 | 0.057 |
| Std | 0.011 | 0.096 | 0.023 |



a – real image



b – classified image

Figure 15 – Example of MSAM classification

5.4.4 Different light conditions

The results (Table 25) show that the *MSAM* algorithm behaves the same under different light conditions, as expected with 94.1% average success and a FPR of 6.8% and FNR of 7.3%.

Table 25 – MSAM – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Strong | 0.941 | 0.062 | 0.073 |
| Medium | 0.941 | 0.073 | 0.073 |
| Low | 0.942 | 0.068 | 0.073 |
| Average | 0.941 | 0.068 | 0.073 |
| Std | 0.000 | 0.006 | 0.000 |

5.4.5 Noise

It shows that noise until 2% does not affect performance. Beyond 2% noise, total success decreases (from 93.5 to 92.1), with higher FPR (21.1 vs. 13.3) and higher FNR (8.6 vs. 7.5)

Table 26 – MSAM – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------------|----------------------------|----------------------------|
| 1% | 0.941 | 0.082 | 0.073 |
| 2% | 0.935 | 0.133 | 0.075 |
| 5% | 0.921 | 0.212 | 0.086 |
| 7% | 0.907 | 0.234 | 0.102 |
| 8% | 0.902 | 0.237 | 0.108 |
| 10% | 0.887 | 0.242 | 0.126 |

5.4.6 Training by different number of hypercubes

As in the *SAM* algorithm, it shows a drastic decrease in performances when using four and five hypercubes as training sets. The total success reduces from 94% to 72%, the FPR increased to 36.9% from 6.8% and the FNR increased to 25.4% from 7.3%. This is expected due to the fact that the *MSAM* algorithm behaves exactly like the *SAM* algorithm with just fewer wavelengths.

Table 27 – MSAM – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| 4 Hypercubes | 0.797 | 0.264 | 0.168 |
| 5 Hypercubes | 0.650 | 0.475 | 0.339 |
| Average | 0.723 | 0.369 | 0.254 |
| Std | 0.104 | 0.149 | 0.121 |

5.5 Channel relation (NDVI)

5.5.1 Independent evaluation of hypercubes

Almost all of the hypercubes received more than 92% total success except for one hypercube which resulted in a low total success of 72% (Hypercube7, Table 28). The average success rate was 93.1%, with 7.2% FPR and 6.4% FNR. The high success rate implies that the algorithm is good for vegetation classification under the same lighting conditions. The high standard deviation in the three measures is due to hypercube7, without it, results are much lower.

Table 28 – NDVI - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| Hypercube1 | 0.965 | 0.043 | 0.028 |
| Hypercube2 | 0.969 | 0.047 | 0.015 |
| Hypercube3 | 0.975 | 0.031 | 0.020 |
| Hypercube4 | 0.976 | 0.010 | 0.034 |
| Hypercube5 | 0.979 | 0.016 | 0.025 |
| Hypercube6 | 0.929 | 0.042 | 0.122 |
| Hypercube7 | 0.725 | 0.314 | 0.203 |
| Average | 0.931 | 0.072 | 0.064 |
| Std | 0.092 | 0.108 | 0.072 |

5.5.2 Cross Validation

The cross validation results (Table 29) indicates very good classifications results (high total success of 95.9% and low FPR and FNR of 4% and 4.1% respectfully). This implies that the algorithm can classify well, when the training and testing data are with the same lighting conditions.

Table 29 – NDVI – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------|---------------------|
| 0.959 | 0.040 | 0.041 |

5.5.3 Training by two hypercubes

The high total success criteria shows that even when hypercubes are taken in different scenarios with different light conditions, there is a common element between all of the hypercubes (Table 30). All of the measures (total success, FPR and the FNR) are approximately the same (low standard deviation) which implies that the algorithm is robust to the training set.

Table 30 – NDVI – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| Training Set 1 | 0.968 | 0.077 | 0.029 |
| Training Set 2 | 0.967 | 0.062 | 0.037 |
| Average | 0.968 | 0.069 | 0.033 |
| Std | 0.001 | 0.011 | 0.006 |



a – real image



b – classified image

Figure 16 – Example of NDVI classification

5.5.4 Different light conditions

The results (Table 31) indicate that the *NDVI* algorithm behaves the same under different light conditions, as expected with average success of 96.8%, 7.6% FPR and 2.9% FNR.

Table 31 – NDVI – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Strong | 0.968 | 0.077 | 0.029 |
| Medium | 0.968 | 0.081 | 0.029 |
| Low | 0.969 | 0.071 | 0.029 |
| Average | 0.968 | 0.076 | 0.029 |
| Std | 0.000 | 0.005 | 0.000 |

5.5.5 Noise

It shows that noise until 2% does not affect performance. There is a small decrease in performances with noises up to 5%, resulting in higher FPR (28.6 vs. 20.5) and FNR (3.4 vs. 3.1) and lower total success (94.6 vs. 95.9). Performance greatly decreases with noise above 5% (Table 32).

Table 32 – NDVI – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------------|----------------------------|----------------------------|
| 1% | 0.965 | 0.126 | 0.030 |
| 2% | 0.959 | 0.205 | 0.031 |
| 5% | 0.946 | 0.286 | 0.034 |
| 7% | 0.936 | 0.323 | 0.038 |
| 8% | 0.934 | 0.329 | 0.039 |
| 10% | 0.927 | 0.344 | 0.044 |

5.5.6 Training by different number of hypercubes

Results indicate only small decrease in performance when using four and five hypercubes as training sets. The total success reduces from 96.8% to 94.2%, the FPR decreased to 4.4% from 7.6% and the FNR increased to 5.7% from 2.9%. This suggests that adding hypercubes to the training data decreases the overall performances, but only moderately.

Table 33 – NDVI – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| 4 Hypercubes | 0.954 | 0.042 | 0.042 |
| 5 Hypercubes | 0.931 | 0.044 | 0.071 |
| Average | 0.942 | 0.043 | 0.057 |
| Std | 0.016 | 0.001 | 0.020 |

5.6 Channel Relations Tree (NDVI_T)

5.6.1 Independent evaluation of hypercubes

Almost all of the hypercubes resulted in more than 92% total success except for one hypercube which resulted in a low total success of 80.4% (Hypercube7) with high FPR and FNR of 22.4% and 14.4% respectively (Table 34).

The average success rate was 94.9%, 6.4% FPR and 3.5% FNR. The high success rate implies the algorithm is good for vegetation classification under the same lighting conditions. The high standard deviation in the three measures is due to hypercube7, without it, the standard deviation is much lower.

Table 34 – NDVI_T - Independent evaluation of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Hypercube1 | 0.968 | 0.052 | 0.015 |
| Hypercube2 | 0.972 | 0.050 | 0.006 |
| Hypercube3 | 0.977 | 0.033 | 0.014 |
| Hypercube4 | 0.978 | 0.019 | 0.023 |
| Hypercube5 | 0.989 | 0.020 | 0.004 |
| Hypercube6 | 0.955 | 0.050 | 0.036 |
| Hypercube7 | 0.804 | 0.224 | 0.144 |
| Average | 0.949 | 0.064 | 0.035 |
| Std | 0.065 | 0.072 | 0.049 |

5.6.2 Cross Validation

The cross validation results (Table 35) indicate very good classifications results (high total success of 96.5% and low FPR and FNR of 7.8% and 1.7% respectfully). This implies that the algorithm can classify well, when the training and testing data are with the same lighting conditions.

Table 35 – NDVI_T – Cross Validation

| Total Success Rate | False Positive Rate | False Negative Rate |
|---------------------------|----------------------------|----------------------------|
| 0.965 | 0.078 | 0.017 |

5.6.3 Training by two hypercubes

The algorithm has good classification results (Table 36). There is a big difference between the two training sets in the total success and FPR while the FNR shows approximately the same results. This suggests that the algorithm is very sensitive to different types of training sets

Table 36 – NDVI_T – Training by two hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| Training Set 1 | 0.920 | 0.362 | 0.024 |
| Training Set 2 | 0.968 | 0.064 | 0.030 |
| Average | 0.944 | 0.213 | 0.027 |
| Std | 0.034 | 0.211 | 0.004 |



a – real image



b – classified image

Figure 17 – Example of NDVI_T classification

5.6.4 Different light conditions

The results (Table 37) indicate that the *NDVI_T* algorithm behaves the same under different light conditions, as expected with average success of 92.2% and a FPR of 35.7% and FNR of 2.3%.

Table 37 – NDVI_T – Different light conditions

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|---------------------------|----------------------------|----------------------------|
| Strong | 0.920 | 0.362 | 0.024 |
| Medium | 0.921 | 0.357 | 0.023 |
| Low | 0.925 | 0.351 | 0.023 |
| Average | 0.922 | 0.357 | 0.023 |
| Std | 0.003 | 0.005 | 0.001 |

5.6.5 Noise

It shows that noise until 2% does not affect performance and results in 92% total success. There is a small decrease in performances with noises over 5% (Table 20) resulting in 90.9% success and a small increase in the FPR (to 39%) and in the FNR (to 3.3%).

Table 38 – NDVI_T – Noise

| Noise Level | Total Success Rate | False Positive Rate | False Negative Rate |
|--------------------|---------------------------|----------------------------|----------------------------|
| 1% | 0.920 | 0.363 | 0.025 |
| 2% | 0.921 | 0.364 | 0.026 |
| 5% | 0.915 | 0.379 | 0.029 |
| 7% | 0.909 | 0.391 | 0.033 |
| 8% | 0.907 | 0.395 | 0.034 |
| 10% | 0.902 | 0.404 | 0.038 |

5.6.6 Training by different number of hypercubes

The performances with five and four hypercubes are compared to the average of the two hypercubes. The total success reduces from 94.4% to 92.7%, the FPR decreased to 10.6% from 21.3% and the FNR increased to 2.9% from 2.7%.

Table 39 – NDVI_T – Training by different number of hypercubes

| | Total Success Rate | False Positive Rate | False Negative Rate |
|----------------|--------------------|---------------------|---------------------|
| 4 Hypercubes | 0.923 | 0.121 | 0.019 |
| 5 Hypercubes | 0.931 | 0.090 | 0.039 |
| Average | 0.927 | 0.106 | 0.029 |
| Std | 0.005 | 0.022 | 0.014 |

5.7 Dynamic channel relation

For *NDVI*, the two wavelengths that were chosen (from the regular channel relation) were 800 and 665 nm. Figure 18 presents the total success of the entire 200 simulations (all of the possibilities of the two weights). In relation to [Formula 18], α is the weight of the 800 nm and β is the weight of the 665 nm. It can be seen that when $\alpha = \beta$ in the ranges of 0.2-1, the “total success” is the same, as expected due to the fact that the weights cancel each other [Formula 19]. The reason that this phenomena is not continuous when α and β is greater than 1 is that the intensity reaches saturation, and then the intensity equals the saturation level and does not reach the actual multiplication level. It can be seen that best success is obtained for equal α and β with small values. This fact provides the information that when capturing both images (665 and 800 nm), the exposure times must be identical and small enough so as to ensure the image is not saturated (especially the 800 nm image).

$$[\text{Formula 19}] \quad \text{Index} = \frac{\alpha C_{\text{One}} - \beta C_{\text{Two}}}{\alpha C_{\text{One}} + \beta C_{\text{Two}}} = \frac{\alpha C_{\text{One}} - \alpha C_{\text{Two}}}{\alpha C_{\text{One}} + \alpha C_{\text{Two}}} = \frac{C_{\text{One}} - C_{\text{Two}}}{C_{\text{One}} + C_{\text{Two}}}$$

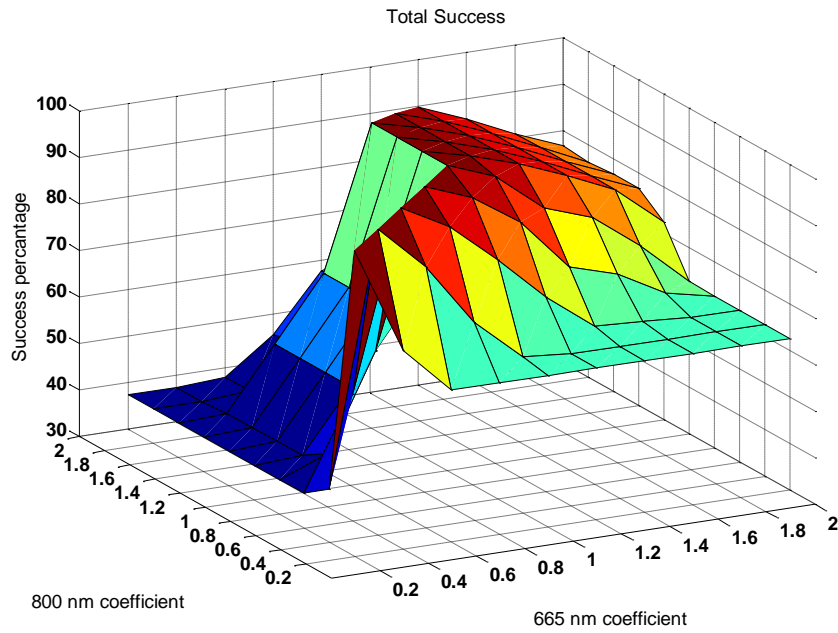


Figure 18 – Total success in dynamic channel relation algorithm

The False Positive Rate and False Negative Rate (Figure 19 and Figure 20), are connected, and choosing a low level of one of them, implies that the other shall be high. When α and β are equal and are less or equal to 1, the sum of False Positive Rate and type 2 errors are the smallest (this also supports the total success).

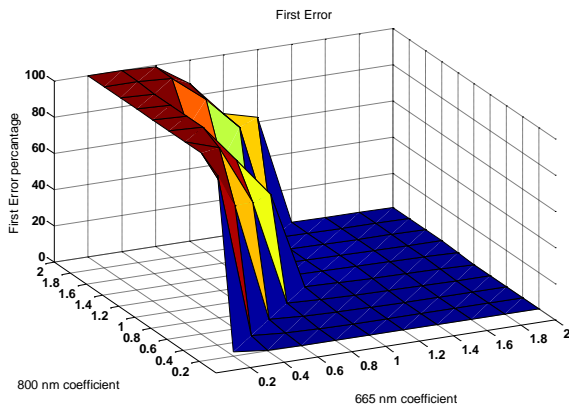


Figure 19 - False positive rate in dynamic channel relation algorithm

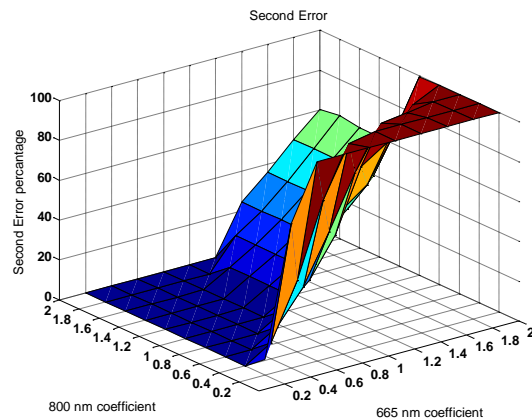


Figure 20 - False negative rate in dynamic channel relation algorithm

5.8 Cloud interference

The difference between the normalized pixels with the direct sunlight and the ones with the clouds are shown in Figure 21. It shows that the differences are not big (maximum difference is 0.055), but still, some wavelengths are better to use in respect to equality toward sunlight and clouds, because they behave the same. For example, the wavelength in 680 nm has a difference value of 0, while the wavelength in 665 nm has a difference value of 0.033.

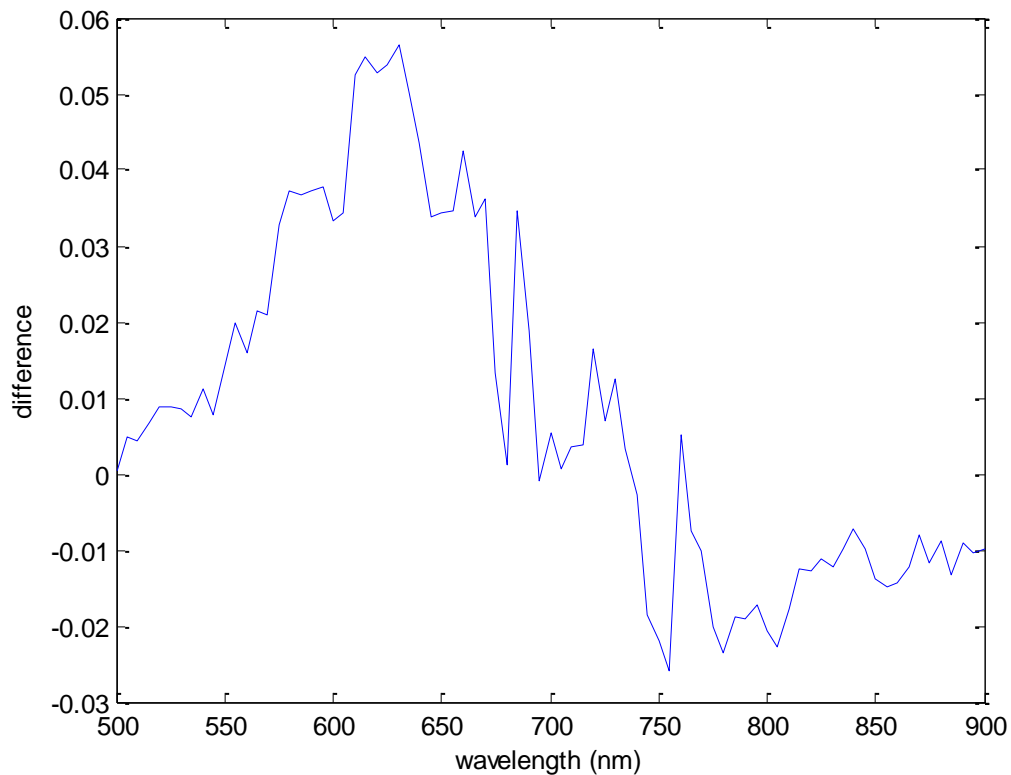


Figure 21 – Normalized difference between direct sunlight and clouds

5.9 Discussion

Seven algorithms have been evaluated for classifying between vegetation and non-vegetation (Table 40). All of the algorithms had a slight decrease in performance when the data had low noises (up to 5%). The *MSAM* algorithm provided very close performances as compared to the *SAM* algorithm, which implies that most of the important behavior of vegetation lies in a few wavelengths, while the remaining wavelengths add little information. The *NDVI_T* surprisingly had lower performances than the *NDVI*; this may be because the decision tree had over-fitting to a special kind of vegetation. When changing the training data, the decision tree received similar performances to the *NDVI* algorithm. The *MSAM* which uses five wavelengths had lower performance as compared to the *NDVI* which uses only two wavelengths; this implies that the number of wavelengths does not directly influence performance. The *SAM* and the *NDVI* algorithms were robust for different training sets with only two hypercubes used in training. The FPR is higher than the FNR due to the fact that there were more pixels of non-vegetation in the training data than vegetation pixels. If the FPR is more important than the FNR, then weights can be added to the vegetation pixels in the training stage while constructing the decision tree. All of the wavelengths behave approximately the same under clouds and direct sunlight, but still, some wavelengths are without any difference at all.

Table 40 – Comparison between the different algorithms

| Algorithm Type | Total Success | False Positive Rate | False Negative Rate | Number of wavelengths used |
|-------------------------------|---------------|---------------------|---------------------|----------------------------|
| Mahalanobis distance | 0.701 | 0.586 | 0.057 | 81 |
| Derivative | 0.871 | 0.460 | 0.032 | 9 |
| SAM | 0.973 | 0.059 | 0.023 | 81 |
| MSAM | 0.949 | 0.130 | 0.057 | 5 |
| Channel Relation (NDVI) | 0.968 | 0.069 | 0.033 | 2 |
| Channel Relation Tree(NDVI_T) | 0.943 | 0.212 | 0.030 | 3 |

Chapter Six: Conclusions and future work

6.1 Conclusions

In this thesis different algorithms were developed and tested for vegetation classification in outdoor conditions with unknown light conditions. Seven algorithms were evaluated (including variations of known algorithms). The algorithm's parameters were selected by a known decision tree algorithm.

The best algorithms that were found for the vegetation classification problem were *SAM* and *NDVI*. The reason that the *NDVI* algorithm was considered better is that it uses only two wavelengths. The *SAM* algorithm has slightly better results but used 80 wavelengths. For near real-time application, the *SAM* algorithm is preferable. The *NDVI* algorithm is best fit for real-time operation.

6.2 Future work

The algorithms presented in this work can be employed in real-time on a mobile robot for navigation in outdoor surroundings. However, there are several areas that remain open for future work.

Spatial Dimension

In this research, all of the algorithms evaluated each pixel by itself. Since the hypercube data comes from the real world, there is a likely connection between adjunction pixels. A better classification algorithm can be developed by using this connection in order to increase performance.

Performance measures

Other performance measures can be defined, in order to take into considerations the spatial aspect of the hypercube instead of looking at each pixel only. In this research, each performance measure was considered separately, however, a unified measure can be developed by combining TS, FNR and FPR using weights.

Other materials

To enable autonomous robots to perform in real world environment, improved environment recognition is necessary. This can be achieved by classifying additional materials like asphalt, water, mud with the methodology presented in this work.

Chapter Seven: References

- [1] Xing J., Saeys W. and De Baerdemaeker J., 2006, Combination of chemometric tools and image processing for bruise detection on apples, *Computer and Electronics in Agriculture* 56: 1-13.
- [2] Okamoto H., Murata T., Kataoka T. and Hata S.H., 2007, Plant classification for weed detection using hyperspectral imaging with wavelet analysis, *Weed Biology and Management* 7: 31-37.
- [3] Xing J., Jancsok P. and De Baerdemaeker J., 2007, Stem-end/Calyx Identification on Apples using Contour Analysis in Multispectral Images, *Biosystems Engineering* 96(2): 231-237.
- [4] Park B., Windham W. R., Lawrence K. C. and Smith D. P., 2007, Contaminant Classification of Poultry Hyperspectral Imagery using a Spectral Angle Mapper Algorithm, *Biosystems Engineering* 96(3): 323-333.
- [5] Ye X., Sakai K., Okamoto H. and Garciano L. O., 2008, A ground-based hyperspectral imaging system for characterizing vegetation spectral features, *Computer and Electronics in Agriculture* 63: 13-21.
- [6] Alchanatis V., Ridet L., Hetzroni A. and Yaroslavsky L., 2005, Weed detection in multi-spectral images of cotton fields, *Computer and Electronics in Agriculture* 47: 243-260.
- [7] Naganathan G. K., Grimes L. M., Subbiah J., Calkins C. R., Samal A. and Meyer G. E., 2008, Visible/near-infrared hyperspectral imaging for beef tenderness prediction, *Computer and Electronics in Agriculture* 64: 225-233.
- [8] Lefcote A. M., Kim M. S., Chen Y. R. and Kang S., 2006, Systematic approach for using hyperspectral imaging data to develop multispectral imaging systems: Detection of feces on apples, *Computer and Electronics in Agriculture* 54: 22-35.
- [9] Fogler R. J., 2003, Multi and Hyper-Spectral sensing for Autonomous Ground Vehicle Navigation, Sandia National Laboratories, Livermore, California 94550.
- [10] Almog O., 2008, Wavelet decomposition for reducing flux density effects on hyperspectral classification, Ph.D. thesis. Technion, Haifa, Israel 32000.
- [11] Tomiya M. and Ageishi A., 2003, Registration of remote sensing image data based on wavelet transform, *ISPRS Archives*, XXXIV(3) - W8, Munich, 17-19.
- [12] Gong, P., Yu, B., 2001, Conifer species recognition: effects of data transformation, *International Journal on Remote Sensing* 22(17): 3471-3481.
- [13] Bradley D. M., Thayer S. M., Stentz A. and Rander P., 2004, Vegetation detection for mobile robot navigation, Robotics Institute, Carnegie Mellon University, CMU-RI-TR-04-12.
- [14] Shaw G., Manolakis D., 2002, Signal processing for hyperspectral image exploitation, *IEEE Signal Processing Magazine*, 19(1): 12-16.
- [15] Cho M. A. and Skidmore A. K., 2006, A new technique for extracting the red edge position from hyperspectral data: the linear extrapolation method, *Remote Sensing of Environment* 101: 181-193.
- [16] Van der Meer F., 2004, Analysis of spectral absorption features in hyperspectral imagery, *International Journal of Applied Earth Observation and Geoinformation*, 5: 55-68.
- [17] Landgrebe D., 2002, Hyperspectral image data analysis as a high dimensional signal processing problem, *Special Issue of the IEEE Signal Processing Magazine*, 19(1): 17-28.
- [18] Solar Radiation Hand Book, 2008, A joint project of solar energy center, MNRE, Indian Metrological Department.
- [19] Yang C. C., Prasher S. O., Enright P., Madramootoo C., Burgess M., Goel P. K., Callum I., 2003, Application of decision tree technology for image classification using remote sensing data, *Agriculture Systems* 76: 1101-1117.

- [20] Demir B. and Erturk S., 2008, Improved classification and segmentation of hyperspectral images using spectral warping, *International Journal of Remote Sensing*, 29(12): 3657–3663.
- [21] DeVries R. J., 2003, *Spatial Modelling using the Mahalanobis Statistic: Two Examples from the Discipline of Plant Geography*, School of Environmental and Life Sciences, University of Newcastle.
- [22] Short Nicholas, *Remote Sensing Tutorial*, NASA, <http://rst.gsfc.nasa.gov/>.
- [23] Bradley D. M., Unnikrishnan R., Bagnell J., 2007, *Vegetation detection in driving In complex environments*, Robotics Institute, Carnegie Mellon University.
- [24] Yuan J, Sun K and Niu Z, 2010, Vegetation water content estimation using Hyperion hyperspectral data, 18th International Conference on Geoinformatics.
- [25] Jin J, Jiang H, Wang Y and Zhang X, 2010, Temporal and spatial variations of NDVI in Lake Baikal, 18th International Conference on Geoinformatics.
- [26] USGS spectral library, <http://speclab.cr.usgs.gov/spectral-lib.html>.
- [27] Rutzinger M., Pratihast A. K., Elberink S. O. and Vosselman G., 2010, Detection and modeling of 3D trees from mobile laser scanning data, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXVIII, Part 5.
- [28] Chen Y., Raikkonen E., Kaasalainen S., Suomalainen J., Hakala T., Hyyppä J. and Chen R, 2010, Two channel hyperspectral lidar with supercontinuum laser source, *Sensors* 10: 7057-7066.
- [29] Antonucci F., Menesatti P., Canali E., Giorgi S., Maienza A. and Stazi S. R., 2010, Hyperspectral imaging characterization of agricultural topsoil copper concentration, CIGR workshop on image analysis in agriculture, Budapest.

Appendices

Appendix A Software

There were several Matlab scripts and functions written in this thesis.

Some of the functions and scripts are written below.

Export 2clementine – Load data from images and create txt file for Clementine

Check algorithm – displays a classified image according to a chosen algorithm

Mahalanobis algorithm – Perform the Mahalanobis algorithm

Derivative algorithm – Perform the Derivative algorithm

SAM algorithm – Perform the SAM algorithm

MSAM algorithm – Perform the MSAM algorithm

NDVI algorithm – Perform the NDVI algorithm

NDVI_T algorithm – Perform the NDVI_T algorithm

Export 2clementine

```
%Script Name: Export 2clementine
%loads data and export it to a txt file for clementine to read

%%Load the data

[FileName,PathName] = uigetfile('*.txt','Choose the file');
eval(['load ',PathName,FileName, ';'])
eval(['handles.ExposerTime=',FileName(1,1:end-4), ';'])
eval(['clear ', FileName(1,1:end-4), ';'])
handles.PathName=PathName;
handles.FileName=FileName;
%

% Find how many images exist
WaveLength=find(handles.ExposerTime>0)+499;
WaveLength=WaveLength';
handles.WaveLength=WaveLength;
%

% finding the name of the images
LocationTemp=find(FileName=='_');
l=length(LocationTemp);
for i=2:l %identifying the end position of the images name
    if (LocationTemp(i)-LocationTemp(i-1))==1
        Location=LocationTemp(i);
        break
    end
end
clear l LocationTemp
handles.NameLocation=Location;
ImagesName=FileName(1:Location);
%

% load all the images
clear handles.Images
l=length(WaveLength);
for i=1:l

eval(['handles.Images.Image1__',num2str(WaveLength(1,i)), '=', 'imread(''',PathName,ImagesName,num2str(WaveLength(1,i)), '.jpg') ;'])
end
%
eval(['VegImage=imread(''',PathName,ImagesName, '000.jpg') ;'])
[m,n]=size(VegImage);
DistricVegImage=zeros(m,n/3, 'uint16')+1000;
temp=find(VegImage(:, :,1)<100 & VegImage(:, :,2)>200 & VegImage(:, :,3)<100);
DistricVegImage(temp)=1001; % 1001=Vegetation
temp=find(VegImage(:, :,1)<100 & VegImage(:, :,2)<100 & VegImage(:, :,3)>200);
DistricVegImage(temp)=1002; % 1002=Not Declared

%% Write to File

eval(['cd ',PathName])
% fid=fopen('SelfData.txt','w');
```

```

fid=fopen('temp.txt','w');

% Write The header
tic
fprintf(fid,'material ');
for i=1:length(handles.WaveLength)
    fprintf(fid,'%d ',handles.WaveLength(i));
%     fprintf(fid, ',');
end
status = fseek(fid,-1,'cof');
fprintf(fid,'\n');
toc

% Making The Data
[m,n]=size(DistricVegImage);

devide=1;
Mat=zeros(m*n/devide/devide,length(handles.WaveLength)+1,'uint16');
m1=1:devide:m;
n1=1:devide:n;
tempDistric=DistricVegImage(m1,n1);
Mat(:,1)=reshape(tempDistric,[],1);

temp=min(handles.WaveLength):5:max(handles.WaveLength);
tic
for k=1:1:length(temp)
    eval(['temp1=handles.Images.Image1__',num2str(temp(k)),';'])
    temp1=temp1(m1,n1);
    Mat(:,k+1)=reshape(temp1,[],1);

end
toc

PrintToFile='%d ';
for i=1:length(temp)
    PrintToFile=[PrintToFile '%d '];
end
PrintToFile=PrintToFile(1:end-1);
Mat=Mat';

temp=find(Mat(1,:)==1000 | Mat(1,:)==1001);
NewMat=Mat(:,temp);
tic
eval(['fprintf(fid,','',PrintToFile,'\n',NewMat);'])
toc

fclose(fid);
Check algorithm

%Script name: Check algorithm
% This script display classified image according to the user choice of
% hypercube

%%Load the data
load VegData.mat

```

```

[FileName,PathName] = uigetfile('*.txt','Choose the file');
eval(['load ',PathName,FileName, ';'])
eval(['handles.ExposerTime=',FileName(1,1:end-4), ';'])
eval(['clear ', FileName(1,1:end-4), ';'])
handles.PathName=PathName;
handles.FileName=FileName;
%

% Find how many images exist
WaveLength=find(handles.ExposerTime>0)+499;
WaveLength=WaveLength';
handles.WaveLength=WaveLength;
%

% finding the name of the images
LocationTemp=find(FileName=='_');
l=length(LocationTemp);
for i=2:l %identifying the end position of the images name
    if (LocationTemp(i)-LocationTemp(i-1))==1
        Location=LocationTemp(i);
        break
    end
end
clear l LocationTemp
handles.NameLocation=Location;
ImagesName=FileName(1:Location);
%

% load all the images
clear handles.Images
l=length(WaveLength);
for i=1:l

eval(['handles.Images.Image1__',num2str(WaveLength(1,i)), '=', 'imread(''',PathName,ImagesName,num2str(WaveLength(1,i)), '.jpg'') ;'])
end
%
eval(['VegImage=imread(''',PathName,ImagesName, '000.jpg'') ;'])
[m,n]=size(VegImage);
DistricVegImage=zeros(m,n/3, 'uint16')+1000;
temp=find(VegImage(:, :,1)<100 & VegImage(:, :,2)>200 & VegImage(:, :,3)<100);
DistricVegImage(temp)=1001; % 1001=Vegetation
temp=find(VegImage(:, :,1)<100 & VegImage(:, :,2)<100 & VegImage(:, :,3)>200);
DistricVegImage(temp)=1002; % 1002=Not Declared

%% ReArrange the data

% Making The Data
% [m,n]=size(DistricVegImage);
[m,n]=size(handles.Images.Image1__560);

```

```

Mat=zeros(m*n,length(handles.WaveLength)+1,'uint16');

Mat(:,1)=reshape(DistricVegImage,[],1);

temp=min(handles.WaveLength):5:max(handles.WaveLength);

for k=1:1:length(temp)
    eval(['temp1=handles.Images.Image1__',num2str(temp(k)),';'])

    Mat(:,k+1)=reshape(temp1,[],1);
end

VegType=menu('Choose The Training Picture/s','14 & 15','16 &
09','14','15','16','01','09','22','25');
switch VegType
    case 1
        VegMean=VegMean1415;
        VegStd=VegStd1415;
        VegStdMinorIndex=VegStdMinorIndex1415;

    case 2
        VegMean=VegMean1609;
        VegStd=VegStd1609;
        VegStdMinorIndex=VegStdMinorIndex1609;

    case 3
        VegMean=VegMean14;
        VegStd=VegStd14;
        VegStdMinorIndex=VegStdMinorIndex14;

    case 4
        VegMean=VegMean15;
        VegStd=VegStd15;
        VegStdMinorIndex=VegStdMinorIndex15;

    case 5
        VegMean=VegMean16;
        VegStd=VegStd16;
        VegStdMinorIndex=VegStdMinorIndex16;

    case 6
        VegMean=VegMean01;
        VegStd=VegStd01;
        VegStdMinorIndex=VegStdMinorIndex01;

    case 7
        VegMean=VegMean09;
        VegStd=VegStd09;
        VegStdMinorIndex=VegStdMinorIndex09;

    case 8
        VegMean=VegMean22;
        VegStd=VegStd22;
        VegStdMinorIndex=VegStdMinorIndex22;

```



```

    case 9
        VegMean=VegMean25;
        VegStd=VegStd25;
        VegStdMinorIndex=VegStdMinorIndex25;

end
VegStd=VegStd+1;

LightValue=menu('Choose The Light Intensity Factor','1 - normal','0.9 - Medium','0.7 - low');
if LightValue >1
    switch LightValue
        case 1
            LightValue=1;
        case 2
            LightValue = 0.9;
        case 3
            LightValue = 0.7;
    end
    Mat(:,2:end)=Mat(:,2:end)*LightValue;
end

NoiseValue=menu('Choose The Noise Intensity Value','None','1% - 2','2% - 5','5% - 12','7% - 18','8% - 20','10% - 25');
if NoiseValue>1
    switch NoiseValue
        case 2
            NoiseValue=2;
        case 3
            NoiseValue = 5;
        case 4
            NoiseValue = 12;
        case 5
            NoiseValue = 18;
        case 6
            NoiseValue = 20;
        case 7
            NoiseValue = 25;
    end
    RandMat1=int8(ceil(rand(300000,size(Mat,2)-1)-0.5));
    RandMat2=int8(ceil(rand(size(Mat,1)-300000,size(Mat,2)-1)-0.5));
    RandMat=[RandMat1;RandMat2];
    clear RandMat1 RandMat2
    RandMat(RandMat==0)=-1;
    tempM=Mat(:,2:end);

    tt=find(tempM>(255-NoiseValue) );
    RandMat(tt)=-1;
    tt=find(tempM<(NoiseValue) );
    RandMat(tt)=1;
    RandMat=int8(RandMat*NoiseValue);

    tempM=int16(tempM) + int16(RandMat);
    tempM(tempM>255)=255;
    tempM(tempM<0)=0;
    Mat(:,2:end)=uint16(tempM);

```

```

clear RandMat tempM tt
end

eval(['ImageToPresent1=imread('',PathName,ImagesName,'590.jpg') ;'])
ImageToPresent(:, :, 1)=ImageToPresent1;
ImageToPresent(:, :, 2)=ImageToPresent1;
ImageToPresent(:, :, 3)=ImageToPresent1;
Image11=zeros(m*n,1,'uint16')+100;
tic
for x=1:m*n
if mod(x,1000)==0
    sss=waitbar(x/(m*n));
end
    flag=DT2_2ndvi(Mat(x,2:end),VegType);
%     flag=DT3_1ndvi(Mat(x,2:end),VegType);
%     flag=DT4_Derivative(Mat(x,2:end),VegType);
%     flag=DT5_Mahalanobis(Mat(x,2:end),VegMean,VegStd,VegType);
%     flag=DT6_SAM(Mat(x,2:end),VegMean,VegType);
%     flag=DT7_SAM_Minor(Mat(x,2:end),VegMean,VegStdMinorIndex,VegType);
    if flag==1
        x1=mod(x,m);
        if x1==0
            x1=m;
        end
        y1=ceil(x/m);
        ImageToPresent(x1,y1,1)=0;
        ImageToPresent(x1,y1,2)=255;
        ImageToPresent(x1,y1,3)=0;
    end
end
toc
close(sss)
figure(1)
subplot(1,2,1)
imshow(ImageToPresent)
title('green is the Vegetation')
figure,
subplot(1,2,2)
imshow(ImageToPresent1)
title('Real Picture')

```

Mahalanobis algorithm

```

%Function Name: Mahalanobis algorithm
% Perform the Mahalanobis algorithm
function [ output ] = DT5_Mahalanobis(SpectralVector,VegMean,VegStd,VegType)

```

```

Distance = sum(((double(SpectralVector)-VegMean)./(VegStd)).^2) ;

```

```

if VegType==1

```

```

    if Distance<= 77.426
        output=1;
    end
    if Distance> 77.426
        output=0;
    end
end

if VegType==2
    if Distance<= 2517.488
        output=1;
    end
    if Distance> 2517.488
        output=0;
    end
end

end

end

```

Derivative algorithm

```

%Function Name: Derivative algorithm
% Perform the Derivative algorithm
function [ output ] = DT4_Derivative(SpectralVector,VegType)

der=SpectralVector(2:end)-SpectralVector(1:end-1);
der=double(der)./5;
Step=5;
Start=500;

if VegType==1
    if der(((695-Start)/Step) +1)<=0
        if der(((730-Start)/Step) +1)<=1
            if der(((715-Start)/Step) +1)<=3
                if der(((710-Start)/Step) +1)<=4
                    if der(((730-Start)/Step) +1)<=0
                        output=0;
                    end
                    if der(((730-Start)/Step) +1)>0
                        if der(((605-Start)/Step) +1)<=1
                            if der(((765-Start)/Step) +1)<=1
                                output=0;
                            end
                            if der(((765-Start)/Step) +1)>1
                                output=1;
                            end
                        end
                        if der(((605-Start)/Step) +1)>1
                            output=0;
                        end
                    end
                end
            end
        end
        if der(((710-Start)/Step) +1)>4
            output=1;
        end
    end
    if der(((715-Start)/Step) +1)>3

```

```

        output=1;
    end
end
if der(((730-Start)/Step) +1)>1
    if der(((655-Start)/Step) +1)<=3
        output=1;
    end
    if der(((655-Start)/Step) +1)>3
        output=0;
    end
end
end
if der(((695-Start)/Step) +1)>0
    output=1;
end
end

if VegType==2
    if der(((695-Start)/Step) +1)<=0
        if der(((730-Start)/Step) +1)<=1
            if der(((690-Start)/Step) +1)<=0
                if der(((560-Start)/Step) +1)<=0
                    if der(((730-Start)/Step) +1)<=0
                        output=0;
                    end
                    if der(((730-Start)/Step) +1)>0
                        output=1;
                    end
                end
                if der(((560-Start)/Step) +1)>0
                    output=0;
                end
            end
            if der(((690-Start)/Step) +1)>0
                output=1;
            end
        end
        if der(((730-Start)/Step) +1)>1
            if der(((560-Start)/Step) +1)<=1
                output=1;
            end
            if der(((560-Start)/Step) +1)>1
                output=0;
            end
        end
    end
end
if der(((695-Start)/Step) +1)>0
    if der(((645-Start)/Step) +1)<=2
        output=1;
    end
    if der(((645-Start)/Step) +1)>2
        output=0;
    end
end
end
end
end

```

SAM algorithm

```
%Function Name: SAM algorithm
% Perform the SAM algorithm
function [ output ] = DT6_SAM(SpectralVector,VegMean,VegType)

Distance =
(double(SpectralVector)*VegMean') / (norm(double(SpectralVector))*norm(VegMean)) ;

if VegType==1
    if Distance<= 0.972
        output=0;
    end
    if Distance> 0.972
        output=1;
    end
end

if VegType==2
    if Distance<= 0.965
        output=0;
    end
    if Distance> 0.965
        output=1;
    end
end

end

end
```

MSAM algorithm

```
%Function Name: SAM Minor algorithm
% Perform the MSAM algorithm

function [ output ] = DT7_SAM_Minor(SpectralVector,VegMean,VegStdIndex,VegType)

Distance =
(double(SpectralVector(VegStdIndex))*VegMean(VegStdIndex)') / (norm(double(SpectralVector(VegStdIndex))*norm(VegMean(VegStdIndex)))) ;

if VegType==1
    if Distance<= 0.984
        output=0;
    end
    if Distance> 0.984
        output=1;
    end
end

if VegType==2
    if Distance<= 0.976
        output=0;
    end
    if Distance> 0.976
        output=1;
    end
end
```

```

    end
end

end

```

NDVI algorithm

```

%Function Name: NDVI algorithm
% Perform the NDVI algorithm

function [ output ] = DT3_1ndvi(SpectralVector,VegType)

if VegType == 1
    n665_800= single((SpectralVector(61) - SpectralVector(34)))/
single((SpectralVector(61) + SpectralVector(34)));

    if n665_800 <= 0
        output=0;
    end

    if n665_800 > 0
        output=1;
    end
end

if VegType == 2
    n800_665= single((SpectralVector(61) - SpectralVector(34)))/
single((SpectralVector(61) + SpectralVector(34)));

    if n800_665 <= 0.004
        output=0;
    end

    if n800_665 > 0.004
        output=1;
    end
end

end

end

```

NDVI T algorithm

```

%Function Name: NDVI_T algorithm
% Perform the NDVI_T algorithm

```

```

function [ output ] = DT2_2ndvi(SpectralVector,VegType)

if VegType==1
    n665_740= single((SpectralVector(49) - SpectralVector(34)))/
single((SpectralVector(49) + SpectralVector(34)));
    n665_775= single((SpectralVector(56) - SpectralVector(34)))/
single((SpectralVector(56) + SpectralVector(34)));

    if n665_740 <= 0
        output=0;
    end
    if n665_740 > 0
        if n665_775 <= -0.024
            if n665_775 <= -0.101
                output=0;
            end
            if n665_775 > -0.101
                output=1;
            end
        end
        if n665_775 > -0.024
            output=1;
        end
    end
end

if VegType==2
    n800_665 = single((SpectralVector(61) - SpectralVector(34)))/
single((SpectralVector(61) + SpectralVector(34)));
    n815_665 = single((SpectralVector(64) - SpectralVector(34)))/
single((SpectralVector(64) + SpectralVector(34)));
    n740_665 = single((SpectralVector(49) - SpectralVector(34)))/
single((SpectralVector(49) + SpectralVector(34)));

    if n800_665 <= 0.004
        output=0;
    end
    if n800_665 > 0.004
        if n815_665 <= -0.234
            if n740_665 <= 0.088
                output=0;
            end
            if n740_665 > 0.088
                output=1;
            end
        end
        if n815_665 > -0.234
            output=1;
        end
    end
end
end
end

```

Appendix B Statistical Analysis

1% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.725 | 0.346 | 0.190 |
| 1 | SAM | 0.956 | 0.089 | 0.002 |
| 1 | NDVI | 0.966 | 0.065 | 0.007 |
| 1 | Derivative | 0.724 | 0.383 | 0.006 |
| 1 | NDVI_T | 0.951 | 0.095 | 0.004 |
| 1 | MSAM | 0.894 | 0.053 | 0.138 |
| 2 | Mahalanobis | 0.889 | 0.014 | 0.140 |
| 2 | SAM | 0.979 | 0.031 | 0.016 |
| 2 | NDVI | 0.981 | 0.036 | 0.011 |
| 2 | Derivative | 0.869 | 0.272 | 0.014 |
| 2 | NDVI_T | 0.982 | 0.038 | 0.007 |
| 2 | MSAM | 0.927 | 0.028 | 0.090 |
| 5 | Mahalanobis | 0.832 | 0.029 | 0.201 |
| 5 | SAM | 0.954 | 0.031 | 0.054 |
| 5 | NDVI | 0.906 | 0.035 | 0.117 |
| 5 | Derivative | 0.826 | 0.305 | 0.064 |
| 5 | NDVI_T | 0.912 | 0.032 | 0.109 |
| 5 | MSAM | 0.905 | 0.023 | 0.122 |
| 6 | Mahalanobis | 0.929 | 0.707 | 0.010 |
| 6 | SAM | 0.977 | 0.099 | 0.022 |
| 6 | NDVI | 0.985 | 0.151 | 0.011 |
| 6 | Derivative | 0.884 | 0.785 | 0.005 |
| 6 | NDVI_T | 0.897 | 0.750 | 0.000 |
| 6 | MSAM | 0.987 | 0.092 | 0.011 |
| 7 | Mahalanobis | 0.920 | 0.987 | 0.021 |
| 7 | SAM | 0.994 | 0.104 | 0.004 |
| 7 | NDVI | 0.987 | 0.341 | 0.006 |
| 7 | Derivative | 0.856 | 0.940 | 0.014 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-----------|---------------|---------------------|---------------------|
| 7 | NDVI_T | 0.854 | 0.901 | 0.006 |
| 7 | MSAM | 0.989 | 0.214 | 0.007 |

2% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.729 | 0.341 | 0.189 |
| 1 | SAM | 0.958 | 0.084 | 0.003 |
| 1 | NDVI | 0.960 | 0.076 | 0.008 |
| 1 | Derivative | 0.663 | 0.431 | 0.017 |
| 1 | NDVI_T | 0.944 | 0.108 | 0.005 |
| 1 | MSAM | 0.894 | 0.054 | 0.137 |
| 2 | Mahalanobis | 0.887 | 0.014 | 0.143 |
| 2 | SAM | 0.977 | 0.030 | 0.019 |
| 2 | NDVI | 0.979 | 0.040 | 0.011 |
| 2 | Derivative | 0.787 | 0.381 | 0.031 |
| 2 | NDVI_T | 0.981 | 0.043 | 0.007 |
| 2 | MSAM | 0.918 | 0.049 | 0.095 |
| 5 | Mahalanobis | 0.831 | 0.029 | 0.203 |
| 5 | SAM | 0.952 | 0.031 | 0.056 |
| 5 | NDVI | 0.902 | 0.043 | 0.119 |
| 5 | Derivative | 0.775 | 0.372 | 0.081 |
| 5 | NDVI_T | 0.908 | 0.038 | 0.113 |
| 5 | MSAM | 0.893 | 0.061 | 0.125 |
| 6 | Mahalanobis | 0.931 | 0.704 | 0.011 |
| 6 | SAM | 0.977 | 0.095 | 0.023 |
| 6 | NDVI | 0.979 | 0.301 | 0.011 |
| 6 | Derivative | 0.896 | 0.777 | 0.008 |
| 6 | NDVI_T | 0.904 | 0.738 | 0.001 |
| 6 | MSAM | 0.985 | 0.139 | 0.011 |
| 7 | Mahalanobis | 0.922 | 0.986 | 0.021 |
| 7 | SAM | 0.994 | 0.099 | 0.005 |
| 7 | NDVI | 0.976 | 0.566 | 0.006 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|------------|---------------|---------------------|---------------------|
| 7 | Derivative | 0.878 | 0.923 | 0.012 |
| 7 | NDVI_T | 0.869 | 0.892 | 0.006 |
| 7 | MSAM | 0.985 | 0.360 | 0.008 |

5% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.752 | 0.305 | 0.192 |
| 1 | SAM | 0.961 | 0.068 | 0.014 |
| 1 | NDVI | 0.941 | 0.108 | 0.012 |
| 1 | Derivative | 0.602 | 0.472 | 0.110 |
| 1 | NDVI_T | 0.919 | 0.149 | 0.008 |
| 1 | MSAM | 0.886 | 0.063 | 0.145 |
| 2 | Mahalanobis | 0.873 | 0.013 | 0.158 |
| 2 | SAM | 0.959 | 0.016 | 0.053 |
| 2 | NDVI | 0.974 | 0.052 | 0.013 |
| 2 | Derivative | 0.727 | 0.444 | 0.039 |
| 2 | NDVI_T | 0.975 | 0.056 | 0.009 |
| 2 | MSAM | 0.894 | 0.063 | 0.122 |
| 5 | Mahalanobis | 0.820 | 0.024 | 0.215 |
| 5 | SAM | 0.943 | 0.029 | 0.069 |
| 5 | NDVI | 0.891 | 0.057 | 0.128 |
| 5 | Derivative | 0.738 | 0.410 | 0.110 |
| 5 | NDVI_T | 0.897 | 0.059 | 0.120 |
| 5 | MSAM | 0.867 | 0.121 | 0.138 |
| 6 | Mahalanobis | 0.941 | 0.693 | 0.016 |
| 6 | SAM | 0.973 | 0.093 | 0.026 |
| 6 | NDVI | 0.966 | 0.497 | 0.013 |
| 6 | Derivative | 0.882 | 0.804 | 0.009 |
| 6 | NDVI_T | 0.904 | 0.742 | 0.002 |
| 6 | MSAM | 0.977 | 0.298 | 0.015 |
| 7 | Mahalanobis | 0.930 | 0.986 | 0.020 |
| 7 | SAM | 0.991 | 0.082 | 0.008 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|------------|---------------|---------------------|---------------------|
| 7 | NDVI | 0.960 | 0.716 | 0.007 |
| 7 | Derivative | 0.871 | 0.923 | 0.012 |
| 7 | NDVI_T | 0.881 | 0.888 | 0.007 |
| 7 | MSAM | 0.979 | 0.514 | 0.008 |

7% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.810 | 0.022 | 0.224 |
| 1 | SAM | 0.916 | 0.026 | 0.107 |
| 1 | NDVI | 0.883 | 0.066 | 0.135 |
| 1 | Derivative | 0.731 | 0.417 | 0.119 |
| 1 | NDVI_T | 0.888 | 0.076 | 0.127 |
| 1 | MSAM | 0.852 | 0.120 | 0.158 |
| 2 | Mahalanobis | 0.785 | 0.242 | 0.193 |
| 2 | SAM | 0.950 | 0.057 | 0.045 |
| 2 | NDVI | 0.920 | 0.141 | 0.019 |
| 2 | Derivative | 0.577 | 0.486 | 0.208 |
| 2 | NDVI_T | 0.905 | 0.168 | 0.014 |
| 2 | MSAM | 0.866 | 0.074 | 0.168 |
| 5 | Mahalanobis | 0.859 | 0.013 | 0.173 |
| 5 | SAM | 0.910 | 0.016 | 0.115 |
| 5 | NDVI | 0.969 | 0.060 | 0.015 |
| 5 | Derivative | 0.726 | 0.446 | 0.040 |
| 5 | NDVI_T | 0.969 | 0.069 | 0.011 |
| 5 | MSAM | 0.867 | 0.064 | 0.154 |
| 6 | Mahalanobis | 0.951 | 0.668 | 0.021 |
| 6 | SAM | 0.970 | 0.097 | 0.029 |
| 6 | NDVI | 0.955 | 0.595 | 0.013 |
| 6 | Derivative | 0.879 | 0.809 | 0.009 |
| 6 | NDVI_T | 0.903 | 0.750 | 0.004 |
| 6 | MSAM | 0.971 | 0.392 | 0.019 |
| 7 | Mahalanobis | 0.942 | 0.985 | 0.020 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|------------|---------------|---------------------|---------------------|
| 7 | SAM | 0.987 | 0.079 | 0.013 |
| 7 | NDVI | 0.955 | 0.753 | 0.008 |
| 7 | Derivative | 0.869 | 0.926 | 0.012 |
| 7 | NDVI_T | 0.881 | 0.894 | 0.008 |
| 7 | MSAM | 0.979 | 0.518 | 0.009 |

8% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.792 | 0.019 | 0.241 |
| 1 | SAM | 0.896 | 0.025 | 0.131 |
| 1 | NDVI | 0.880 | 0.071 | 0.138 |
| 1 | Derivative | 0.731 | 0.417 | 0.121 |
| 1 | NDVI_T | 0.885 | 0.082 | 0.128 |
| 1 | MSAM | 0.845 | 0.119 | 0.167 |
| 2 | Mahalanobis | 0.798 | 0.179 | 0.216 |
| 2 | SAM | 0.941 | 0.052 | 0.065 |
| 2 | NDVI | 0.914 | 0.149 | 0.022 |
| 2 | Derivative | 0.576 | 0.488 | 0.218 |
| 2 | NDVI_T | 0.901 | 0.173 | 0.017 |
| 2 | MSAM | 0.858 | 0.077 | 0.178 |
| 5 | Mahalanobis | 0.834 | 0.012 | 0.199 |
| 5 | SAM | 0.889 | 0.016 | 0.140 |
| 5 | NDVI | 0.968 | 0.063 | 0.016 |
| 5 | Derivative | 0.727 | 0.445 | 0.041 |
| 5 | NDVI_T | 0.967 | 0.072 | 0.012 |
| 5 | MSAM | 0.857 | 0.066 | 0.165 |
| 6 | Mahalanobis | 0.961 | 0.606 | 0.027 |
| 6 | SAM | 0.969 | 0.105 | 0.030 |
| 6 | NDVI | 0.953 | 0.608 | 0.013 |
| 6 | Derivative | 0.879 | 0.809 | 0.009 |
| 6 | NDVI_T | 0.903 | 0.751 | 0.004 |
| 6 | MSAM | 0.970 | 0.413 | 0.021 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 7 | Mahalanobis | 0.959 | 0.982 | 0.020 |
| 7 | SAM | 0.985 | 0.084 | 0.014 |
| 7 | NDVI | 0.954 | 0.757 | 0.008 |
| 7 | Derivative | 0.869 | 0.926 | 0.012 |
| 7 | NDVI_T | 0.881 | 0.895 | 0.008 |
| 7 | MSAM | 0.980 | 0.508 | 0.009 |

10% Noise

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.774 | 0.110 | 0.271 |
| 1 | SAM | 0.890 | 0.040 | 0.150 |
| 1 | NDVI | 0.900 | 0.164 | 0.034 |
| 1 | Derivative | 0.578 | 0.486 | 0.216 |
| 1 | NDVI_T | 0.889 | 0.185 | 0.029 |
| 1 | MSAM | 0.829 | 0.087 | 0.212 |
| 2 | Mahalanobis | 0.788 | 0.012 | 0.241 |
| 2 | SAM | 0.839 | 0.017 | 0.193 |
| 2 | NDVI | 0.962 | 0.074 | 0.018 |
| 2 | Derivative | 0.729 | 0.442 | 0.041 |
| 2 | NDVI_T | 0.960 | 0.085 | 0.015 |
| 2 | MSAM | 0.832 | 0.076 | 0.191 |
| 5 | Mahalanobis | 0.761 | 0.014 | 0.269 |
| 5 | SAM | 0.844 | 0.022 | 0.191 |
| 5 | NDVI | 0.871 | 0.086 | 0.145 |
| 5 | Derivative | 0.732 | 0.415 | 0.121 |
| 5 | NDVI_T | 0.877 | 0.097 | 0.134 |
| 5 | MSAM | 0.827 | 0.110 | 0.191 |
| 6 | Mahalanobis | 0.967 | 0.273 | 0.032 |
| 6 | SAM | 0.968 | 0.116 | 0.032 |
| 6 | NDVI | 0.950 | 0.630 | 0.014 |
| 6 | Derivative | 0.879 | 0.809 | 0.009 |
| 6 | NDVI_T | 0.902 | 0.757 | 0.005 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 6 | MSAM | 0.967 | 0.457 | 0.023 |
| 7 | Mahalanobis | 0.977 | 0.964 | 0.020 |
| 7 | SAM | 0.982 | 0.093 | 0.018 |
| 7 | NDVI | 0.952 | 0.766 | 0.008 |
| 7 | Derivative | 0.868 | 0.927 | 0.012 |
| 7 | NDVI_T | 0.881 | 0.896 | 0.008 |
| 7 | MSAM | 0.981 | 0.483 | 0.010 |

High Lighting Conditions

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.721 | 0.354 | 0.184 |
| 1 | SAM | 0.955 | 0.090 | 0.002 |
| 1 | NDVI | 0.968 | 0.061 | 0.006 |
| 1 | Derivative | 0.788 | 0.323 | 0.003 |
| 1 | NDVI_T | 0.957 | 0.086 | 0.004 |
| 1 | MSAM | 0.893 | 0.054 | 0.139 |
| 2 | Mahalanobis | 0.897 | 0.015 | 0.131 |
| 2 | SAM | 0.979 | 0.032 | 0.015 |
| 2 | NDVI | 0.984 | 0.028 | 0.010 |
| 2 | Derivative | 0.927 | 0.170 | 0.009 |
| 2 | NDVI_T | 0.985 | 0.032 | 0.007 |
| 2 | MSAM | 0.927 | 0.028 | 0.090 |
| 5 | Mahalanobis | 0.843 | 0.032 | 0.190 |
| 5 | SAM | 0.954 | 0.031 | 0.053 |
| 5 | NDVI | 0.911 | 0.024 | 0.114 |
| 5 | Derivative | 0.863 | 0.247 | 0.057 |
| 5 | NDVI_T | 0.919 | 0.026 | 0.104 |
| 5 | MSAM | 0.906 | 0.023 | 0.120 |
| 6 | Mahalanobis | 0.922 | 0.719 | 0.007 |
| 6 | SAM | 0.977 | 0.098 | 0.022 |
| 6 | NDVI | 0.987 | 0.106 | 0.011 |
| 6 | Derivative | 0.881 | 0.788 | 0.004 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 6 | NDVI_T | 0.888 | 0.764 | 0.000 |
| 6 | MSAM | 0.988 | 0.075 | 0.011 |
| 7 | Mahalanobis | 0.917 | 0.986 | 0.021 |
| 7 | SAM | 0.994 | 0.105 | 0.004 |
| 7 | NDVI | 0.992 | 0.165 | 0.006 |
| 7 | Derivative | 0.856 | 0.940 | 0.014 |
| 7 | NDVI_T | 0.850 | 0.901 | 0.005 |
| 7 | MSAM | 0.991 | 0.132 | 0.007 |

Medium Lighting Conditions

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.721 | 0.354 | 0.184 |
| 1 | SAM | 0.955 | 0.091 | 0.002 |
| 1 | NDVI | 0.968 | 0.061 | 0.006 |
| 1 | Derivative | 0.788 | 0.323 | 0.003 |
| 1 | NDVI_T | 0.957 | 0.086 | 0.004 |
| 1 | MSAM | 0.893 | 0.054 | 0.139 |
| 2 | Mahalanobis | 0.897 | 0.015 | 0.131 |
| 2 | SAM | 0.979 | 0.032 | 0.015 |
| 2 | NDVI | 0.984 | 0.028 | 0.010 |
| 2 | Derivative | 0.927 | 0.170 | 0.009 |
| 2 | NDVI_T | 0.985 | 0.032 | 0.007 |
| 2 | MSAM | 0.927 | 0.028 | 0.090 |
| 5 | Mahalanobis | 0.843 | 0.032 | 0.190 |
| 5 | SAM | 0.954 | 0.031 | 0.053 |
| 5 | NDVI | 0.911 | 0.024 | 0.114 |
| 5 | Derivative | 0.863 | 0.247 | 0.057 |
| 5 | NDVI_T | 0.919 | 0.026 | 0.104 |
| 5 | MSAM | 0.906 | 0.023 | 0.120 |
| 6 | Mahalanobis | 0.955 | 0.621 | 0.019 |
| 6 | SAM | 0.975 | 0.076 | 0.024 |
| 6 | NDVI | 0.985 | 0.141 | 0.011 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 6 | Derivative | 0.880 | 0.790 | 0.004 |
| 6 | NDVI_T | 0.893 | 0.757 | 0.000 |
| 6 | MSAM | 0.986 | 0.128 | 0.011 |
| 7 | Mahalanobis | 0.986 | 0.398 | 0.001 |
| 7 | SAM | 0.996 | 0.095 | 0.002 |
| 7 | NDVI | 0.994 | 0.151 | 0.003 |
| 7 | Derivative | 0.880 | 0.875 | 0.004 |
| 7 | NDVI_T | 0.851 | 0.883 | 0.001 |
| 7 | MSAM | 0.994 | 0.134 | 0.003 |

Low Lighting Conditions

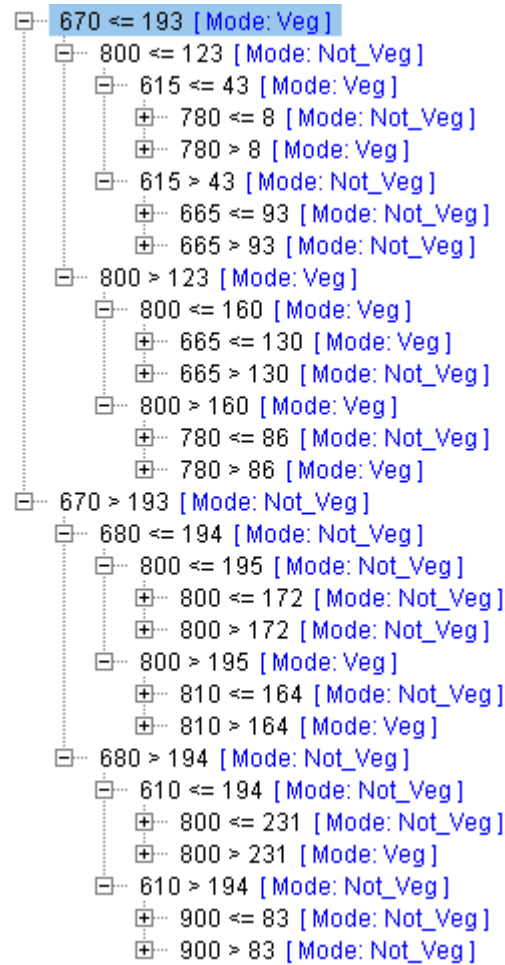
| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 1 | Mahalanobis | 0.721 | 0.354 | 0.184 |
| 1 | SAM | 0.955 | 0.090 | 0.002 |
| 1 | NDVI | 0.968 | 0.061 | 0.006 |
| 1 | Derivative | 0.788 | 0.323 | 0.003 |
| 1 | NDVI_T | 0.957 | 0.086 | 0.004 |
| 1 | MSAM | 0.893 | 0.054 | 0.139 |
| 2 | Mahalanobis | 0.897 | 0.015 | 0.131 |
| 2 | SAM | 0.979 | 0.032 | 0.015 |
| 2 | NDVI | 0.984 | 0.028 | 0.010 |
| 2 | Derivative | 0.927 | 0.170 | 0.009 |
| 2 | NDVI_T | 0.985 | 0.032 | 0.007 |
| 2 | MSAM | 0.927 | 0.029 | 0.090 |
| 5 | Mahalanobis | 0.843 | 0.032 | 0.190 |
| 5 | SAM | 0.954 | 0.031 | 0.053 |
| 5 | NDVI | 0.911 | 0.024 | 0.114 |
| 5 | Derivative | 0.863 | 0.247 | 0.057 |
| 5 | NDVI_T | 0.919 | 0.026 | 0.104 |
| 5 | MSAM | 0.906 | 0.023 | 0.120 |
| 6 | Mahalanobis | 0.963 | 0.910 | 0.034 |
| 6 | SAM | 0.975 | 0.055 | 0.025 |

| Hypercube | Algorithm | Total Success | False Positive Rate | False Negative Rate |
|-----------|-------------|---------------|---------------------|---------------------|
| 6 | NDVI | 0.987 | 0.077 | 0.012 |
| 6 | Derivative | 0.898 | 0.788 | 0.011 |
| 6 | NDVI_T | 0.900 | 0.744 | 0.000 |
| 6 | MSAM | 0.987 | 0.070 | 0.011 |
| 7 | Mahalanobis | 0.980 | 1.000 | 0.020 |
| 7 | SAM | 0.996 | 0.092 | 0.003 |
| 7 | NDVI | 0.994 | 0.162 | 0.003 |
| 7 | Derivative | 0.971 | 0.594 | 0.000 |
| 7 | NDVI_T | 0.868 | 0.869 | 0.000 |
| 7 | MSAM | 0.994 | 0.165 | 0.003 |

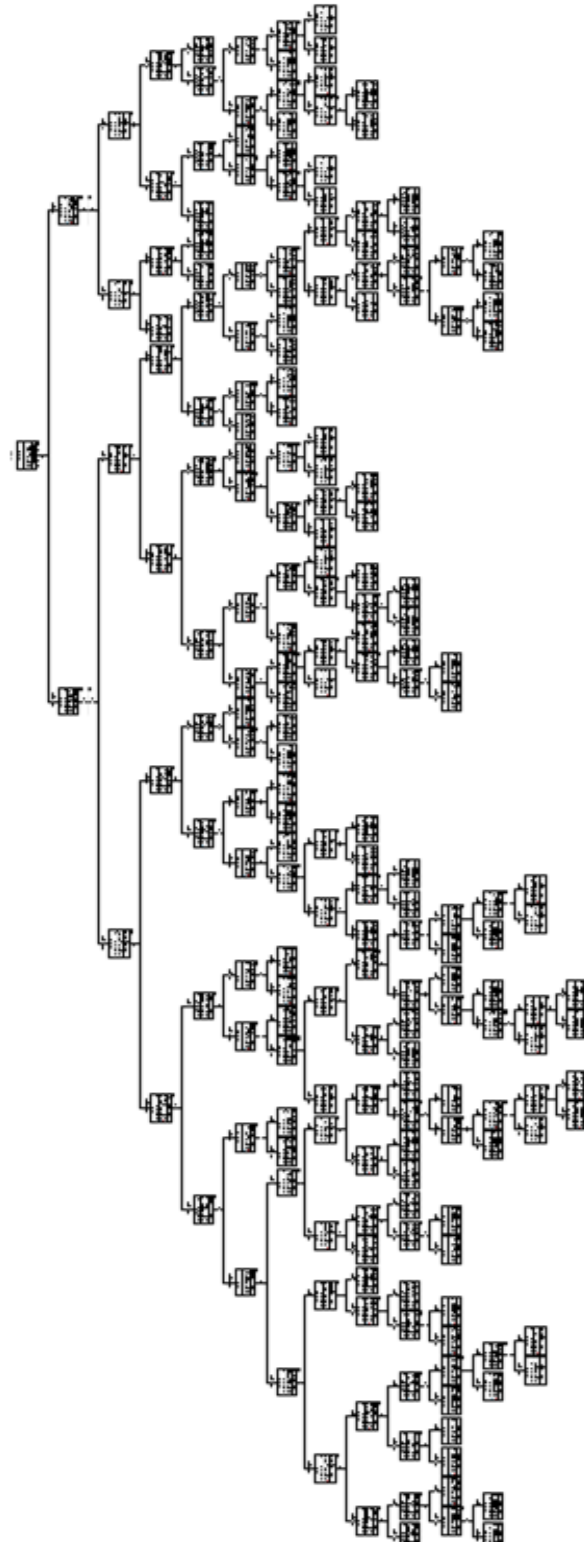
Appendix C Decision Trees

Wavelength Selection

The basic decision tree for wavelength selection is very big, so only the first four levels are displayed here. The first number is the wavelength, and the second number is the intensity level at this specific wavelength.




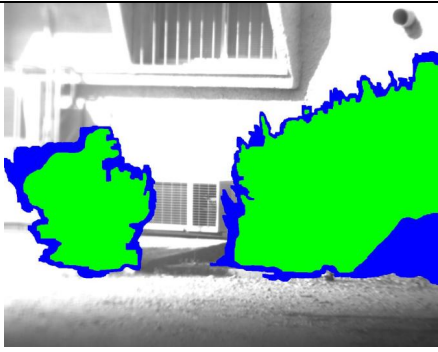

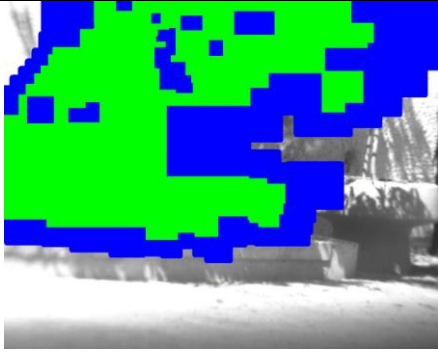

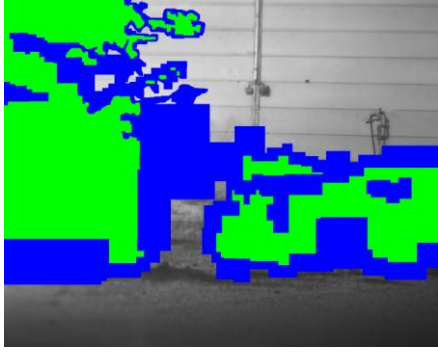
The full decision tree (all the levels) will look like:




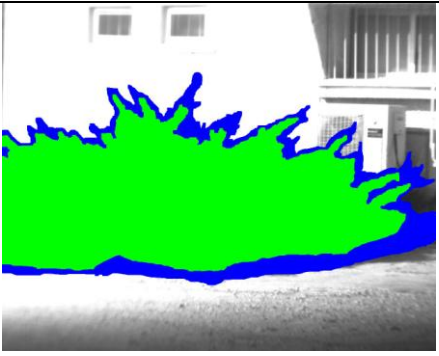

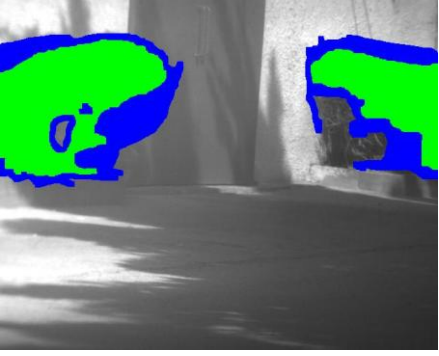




Appendix D Raw Data

Exists on the disk in the *raw data library*

Appendix E manually classified imaged

| | |
|---|--|
|  |  |
|  |  |
|  |  |
| a – image at 590nm | b – manually classified image |

| | |
|---|--|
|  |  |
|  |  |
|  |  |
|  |  |
| <p>a – image at 590nm</p> | <p>b – manually classified image</p> |

פרק חמש : תוצאות ודיון.....28

| | |
|---------|--|
| 28..... | Mahalanobis Distance 5.1 |
| 31..... | אלגוריתמיקה של נגזרות 5.2 |
| 35..... | אלגוריתם מיפוי זווית ספקטראלית 5.3 |
| 38..... | אלגוריתם מיפוי זווית ספקטראלית מוקטן 5.4 |
| 42..... | יחסיות בין ערוצים 5.5 |
| 45..... | עץ של יחסיות בין ערוצים 5.6 |
| 49..... | יחסיות בין ערוצים דינאמית 5.7 |
| 51..... | הפרעה ע"י עננים 5.8 |
| 52..... | דיון 5.9 |

פרק שש : סיכום ועבודה עתידית.....53

| | |
|---------|------------------|
| 53..... | סיכום 6.1 |
| 53..... | עבודה עתידית 6.2 |

פרק שבע:

ביבליוגרפיה.....54

נספחים.....56

| | |
|---------|-------------------------------|
| 56..... | נספח A תוכנה |
| 68..... | נספח B ניתוח סטטיסטי |
| 78..... | נספח C עצי החלטה |
| 80..... | נספח D מידע גולמי |
| 80..... | נספח E תמונות אשר סווגו ידנית |

תוכן עניינים

| | |
|----------|---|
| 1..... | פרק ראשון: הקדמה |
| 1.1..... | 1.1 תיאור הבעיה |
| 1..... | 2.1 מטרות |
| 2..... | 3.1 חידוש המחקר בתזה |
| 3..... | פרק שתיים: סקר ספרות |
| 3..... | 2.1 חישה היפרספקטראלית |
| 4..... | 2.2 אנטראקציה בין קרינת השמש לאובייקט |
| 5..... | 2.3 התנהגות ספקטראלית של צמחייה |
| 6..... | 2.4 אלגוריתמים כלליים של חישה היפרספקטראלית |
| 13..... | פרק שלוש: מתודולוגיה |
| 13..... | 3.1 כללי |
| 13..... | 3.2 תיאור הניסוי |
| 16..... | 3.3 אלגוריתמים |
| 16..... | 3.4 בדיקת הביצועיים |
| 18..... | 3.5 עץ החלטות |
| 19..... | 3.6 כיול |
| 19..... | 3.7 הפרעות ע"י עננים |
| 20..... | 3.5 ניתוחי רגישות |
| 21..... | פרק ארבע: אלגוריתמים |
| 21..... | 4.1 סקירה |
| 21..... | 4.2 Mahalanobis Distance |
| 22..... | 4.3 אלגוריתמיקה של נגזרות |
| 23..... | 4.4 אלגוריתם מיפוי זווית ספקטראלית |
| 23..... | 4.5 אלגוריתם מיפוי זווית ספקטראלית מוקטן |
| 24..... | 4.6 יחסיות בין ערוצים |
| 26..... | 4.7 עץ של יחסיות בין ערוצים |
| 27..... | 4.8 יחסיות בין ערוצים דינאמית |

תקציר

סיווג של צמחייה היא נושא חשוב ב מגוון יישומים כגון ניווט אוטונומי ורובוטים חקלאיים . הנושא חשוב עבור ניווט אוטונומי כיוון שרוב המכשולים מזוהים בעזרת גיאומטריה ו צבעים. מכשולים רכים כגון גבעולים יזוהו כמכשולים לכל דבר וכל הרכב יאלץ לעקוף אותם במקום לנסוע ישירות דרכם , דבר אשר יגרום לנסיעה לא הגיונית . עבור ישומי רובוטיקה חקלאית, הרכב צריך לזהות נתיבים שאפשר לנסוע עליהם , על מנת לא להרוס גידולים חקלאיים , ולכן צריך לדעת להבחין בין צמחייה ללא- צמחייה. לרוב שבילים מזוהים בצורה גיאומטרית על ידי חיישני לייזר ומצלמות צבעוניות. זיהוי זה בעייתי בסביבה פתוחה עם תנאי תאורה משתנים (לדוגמא אור שמש ישיר, עננים...) בשל ההשפעה על המצלמה הצבעונית. היכולת לסווג את החומר שמהווה הדרך ולא רק את הצורה הגיאומטרית שלו והצבע שלו מהווה יכולת עמידה יותר לזיהוי שבילים . מחקר זה מתמקד בסיווג בין צמחייה ללא- צמחייה בתנאי חוץ עם תנאי תאורה לא ידועים.

לצורך מחקר זה , נעשה שימוש במערכת הדמיה היפרספקטראלית אשר רוכשת תמונה ות באורכי גל של 500 עד 900 ננומטר, עם רוחב חצי מקסימום של כ- 5 ננומטר. אזורים שונים צולמו תחת תנאי תאורה שונים (תאורת שמש ישירה , עננים, שעה שונה ביום). האזורים שצולמו מכילים סוגים שונים של צמחייה וחומרים אחרים כגון אספלט, אדמה וקירות של בניינים. שבעה אלגוריתמים פותחו עבור סיווג של צמחייה ולא- צמחייה. 1) Mahalanobis Distance אשר מחשב את Mahalanobis Distance של פיקסל ומשווה אותו ל Mahalanobis Distance של צמחייה שחושב מראש . 2) נגזרות - אשר חישב את הנגזרת בין שני אורכי גל צמודים והשווה אותם לנגזרות של אורכי גל צמודים של צמחייה אשר נמצאו מראש. במקום להשתמש בגל הנגזרות האפשריות , רק הנגזרות החשובות ביותר השו , והם נמצא על ידי עץ החלטות מסוג "5.0 C". 3) מיפוי זווית ספקטראלית - אשר מתחשב בכל אורכי הגל של פיקסל מסוים כווקטור ממימד כל שהוא , ומחשב את הזווית בינו לבין וקטור של צמחייה אשר ידוע מראש . 4) וריאציה של האלגוריתם השלישי - אשר במקום להשתמש בכל אורכי הגל , הוא משתמש בוקטור קטן בהרבה אשר מכיל רק את אורכי הגל החשובים ביותר . אורכי הגל החשובים ביותר נמצאו על ידי עץ החלטות מסוג "5.0 C". 5) אינדקס הפרשי מנורמל של צמחייה - אשר משתמש בשני אורכי גל ומחשב את ההפרש המנורמל ביניהם . על מנת לבחור בצורה אופטימאלית את הפרמטרים של האלגוריתם ואת אורכי הגל הספציפיים, נעשה שימוש בעץ החלטות מסוג "5.0 C". 6) הרחבה של האלגוריתם החמישי - כאשר הוא מחשב מספר אינדקסים שונים , שנוצרו על ידי אורכי גל שונים ומשתמש בעץ החלטות בשביל הסיווג . 7) אדפטציה של האלגוריתם החמישי – כאשר הוא מתחשב בזמן החשיפה של המערכת ההיפרספקטראלית . באלגוריתם זה נבדק מהו זמן החשיפה האופטימאלי עבור כל אורך גל.

התוצאות מראות שהאלגוריתמים "מיפוי זווית ספקטראלית " ו "אינדקס הפרשי מנורמל של צמחייה " הביאו את התוצאות הטובות ביותר מבין שבעת האלגוריתמים , כאשר דיוק הסיווג היה 97% ו96% בהתאמה. שני האלגוריתמים לא הושפעו על ידי תנאי תאורה שונים, תנאי אשר היה הכרחי ובסיסי.

מילות מפתח: הדמיה היפרספקטראלית, סיווג צמחייה

סיווג בין צמחייה ללא-צמחייה בעזרת חישה

היפרספקטראלית

חיבור זה מהווה חלק מהדרישות לקבלת תואר "מגיסטר" בהנדסה

ויצרבין אפריים

מנחים : פרופ' יעל אידן, ד"ר ויקטור אלחנתי

אוניברסיטת בן-גוריון בנגב, הפקולטה למדעי ההנדסה
המחלקה להנדסת תעשייה וניהול

מחבר : ויצרבין אפריים

מנחה : פרופ' יעל אידן

מנחה : ד"ר ויקטור אלחנתי

יושב ראש ועדת הוראה לתואר שני : פרופ' יוסף קרמר

ינואר 2011

סיווג בין צמחייה ללא-צמחייה בעזרת חישה היפרספקטראלית

חיבור זה מהווה חלק מהדרישות לקבלת תואר "מגיסטר" בהנדסה

ויצרבין אפריים

מנחים : פרופ' יעל אידן, ד"ר ויקטור אלחנתי

אוניברסיטת בן-גוריון בנגב, הפקולטה למדעי ההנדסה
המחלקה להנדסת תעשייה וניהול

ינואר 2011