TAMPERE UNIVERSITY OF TECHNOLOGY

LIIMATAINEN, KAISA

Automated detection of structures from remote sensing data

Master of Science Thesis

Examiner: Prof. Olli Yli-Harja
The examiner and the topic were approved in the Computing and Electrical Engineering Faculty Council meeting on June 4^{th} 2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
LIIMATAINEN, KAISA: Automated detection of structures from remote sensing data
Master of Science Thesis, 58 pages, 2 Appendix pages
November 2014
Major: Signal Processing
Examiner: Prof. Olli Yli-Harja
Keywords: Logistic regression, Polynomial modeling, Remote sensing, Drainage network

In just a few decades mire biodiversity has dramatically decreased in southern and central Finland. An objective method for determining the natural state of a mire is needed to direct mire usage for peat mining and agricultural purposes to mires that are already drained beyond restoration. Man-made structures like ditches and roads in and surrounding mires are important descriptors of the natural state. Thus the mire naturality project begins with development of an automated method for detection of these structures, which also is the objective of this thesis.

The development of accurate remote sensing imaging technologies has made it possible to detect small details from the data. National Land Survey of Finland provides high precision LiDAR data and orthophotos that are under open licence and free to use. This data, collected from various mire sites in Southern Ostrobothnia, is used in this thesis. A digital terrain model created from LiDAR data is used for ditch classification and orthophotographs, LiDAR intensity data and digital terrain model are used to detect roads.

The classification is done with logistic regression classifier which selects best features for classification from a large feature set. In ditch detection also polynomial modeling is used to connect broken segments. Logistic regression is a supervised classification method so manually selected coordinate points are used for training the classifier. Different study sites are used for training and testing, and validation is done with manually selected testing points.

Artificial drainage networks were detected well with the method and polynomial modeling improved the results. Ditch points were divided into rough depth classes and the results showed that ditches deeper than half a meter were detected well. Some of the lower ditches were found, too. The percentage of found ditch points from all ditch points (recall) was 90.51 before polynomial modeling and 97.27 after. The road detection accuracy did not correspond to values obtained from ditch detection. The roads were found well but there were many excessive structures that were classified as roads. Yet the ditch detection results indicate that logistic regression classification is a suitable method for this application. For successful classification the feature set needs to be large enough and the training set has to be comprehensive.

Artificial drainage network information will later be used in determining the extent of mire drainage and modeling waterflow patterns. To estimate the impact of drainage, mire type needs to be determined too. This will be done with logistic regression classification of surface texture and height gradient information.

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietotekniikan koulutusohjelma
LIIMATAINEN, KAISA: Automaattinen rakenteiden etsintä kaukokartoitusdatasta
Diplomityö, 58 sivua, 2 liitesivua
Marraskuu 2014
Pääaine: Signaalinkäsittely
Tarkastaja: Prof. Olli Yli-Harja
Avainsanat: Logistinen regressio, Polynomimallinnus, Kaukokartoitus, Ojitusverkosto

Soiden biodiversiteetti on vähentynyt dramaattisesti Etelä- ja Keski-Suomessa muutamassa vuodessa. Jotta turpeennosto ja maatalouden tarpeet voidaan ohjata peruuttamattomasti kuivatetuille soille luonnonvaraisten sijaan, tarvitaan objektiivinen menetelmä suon luonnontilaisuuden analysointiin. Ojat ja tiet suoalueilla kertovat suon luonnontilan heikentymisestä. Näiden rakenteiden automaattinen etsintä on tämän diplomityön tavoite ja ensimmäinen vaihe luonnontilaisuuden automaattisessa määrityksessä.

Kaukokartoitusmenetelmien kehittyminen on mahdollistanut entistäkin pienempien rakenteiden etsinnän kaukokartoitusdatasta. Maanmittauslaitos tarjoaa vapaasti käytettävissä olevia, avoimen aineiston lisenssin alaisia kaukokartoitusaineistoja. Tässä työssä hyödynnetään Maanmittauslaitoksen laserkeilausdataa sekä ortoilmakuvia Etelä-Pohjanmaan soiden alueilta. Ojien luokittelussa käytetään laserkeilausdatasta tehtyä digitaalista maastomallia ja teiden luokittelussa digitaalista maastomallia, ortoilmakuvia sekä laserkeilauksesta saatua intensiteettidataa.

Luokittelu tehdään logistisella regressiolla, jossa suuresta piirrejoukosta valitaan parhaat piirteet kyseiseen luokittelutehtävään. Lisäksi ojien etsinnässä katkenneet ojasegmentit yhdistetään polynomimallinnuksen avulla. Logistinen regressio on ohjattu luokittelumenetelmä, jossa luokittelija opetetaan käsin valituilla koordinaattipisteillä. Opetukseen ja testaukseen käytetään dataa eri suoalueilta, ja tulokset validoidaan käsin valittujen tarkistuspisteiden perusteella.

Ojitusverkostot löytyivät menetelmällä hyvin ja polynomimallinnus paransi tuloksia. Erityisen hyvin luokittelu onnistui puolta metriä syvempien ojien kohdalla mutta ojien madaltuessa myös luokittelutulokset huononivat. Ojapisteistä löydettiin 90.51 prosenttia ennen polynomimallinnusta ja 97.27 prosenttia mallinnuksen jälkeen. Teiden etsintä ei kuitenkaan antanut yhtä hyviä tuloksia. Vaikka tiet löydettiin hyvin, myös monia ylimääräisiä objekteja luokiteltiin teiksi. Ojien etsinnästä saadut tulokset kuitenkin osoittavat, että logistinen regressio on sopiva menetelmä tähän sovellukseen. Luokittelun onnistumiseksi piirrejoukon tulee olla tarpeeksi laaja ja opetusjoukon kattava.

Ojitusverkkoluokittelua tullaan käyttämään suon ojituksen laajuuden arvioinnissa sekä veden virtauksen mallinnuksessa. Jotta ojituksen vaikutuksia voidaan arvioida tarkasti, myös suotyyppi täytyy selvittää. Tämä tullaan tekemään luokittelemalla pintatekstuureja logistisella regressiolla sekä ottamalla huomioon korkeusgradientit.

PREFACE

This master's thesis was carried out in the Department of Signal Processing in Tampere University of Technology. I would like to thank Doctor Pekka Ruusuvuori for supervising this thesis, giving valuable feedback and providing material for realization of this work. Also I want to express my gratitude to my thesis work examiner Professor Olli Yli-Harja.

I am grateful for Docent Raimo Heikkilä for making me familiar with mires of Finland and providing material for environmental aspects of this thesis. In addition I would like to thank Doctor Jari Yli-Hietanen and Assistant Professor Heikki Huttunen.

Tampere, November 14^{th}, 2014

LIIMATAINEN, KAISA

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

μ	Mean
∇	Gradient
$\wedge$	Logical AND
$\neg$	Logical complement
$\vee$	Logical OR
σ	Standard deviation
σ^2	Variance
ALS	Airborne Laser Scanning
CRS	Coordinate Reference System
DEM	Digital Elevation Model
DFT	Discrete Fourier Transform
DSM	Digital Surface Model
DTM	Digital Terrain Model
ETRS	European Terrestrial Reference System
FMF	Flatness Mean Filter
GPS	Global Positioning System
GRS	Geodetic Reference System
ILBP	Improved Local Binary Pattern
ILTP	Improved Local Ternary Pattern
LASSO	Least Absolute Shrinkage and Selection Operator
LBP	Local binary pattern
LiDAR	Light Detection And Ranging (or light radar)
LPF	Lowpass Filter
LTP	Local Ternary Pattern
MBP	Median Binary Pattern
MIT	Massachusetts Institute for Technology
MSE	Mean Square Error
NLS	National Land Survey of Finland (Finnish: Maanmittauslaitos)
RLBP	Robust Local Binary Pattern
STD	Standard Deviation
TM	Transverse Merchator projection
UTM	Universal Transverse Mercator coordinate system

1. INTRODUCTION

Mire biodiversity has dramatically decreased in southern and central Finland in just a few decades. This is due to peatland forestry, agriculture and peat mining. Pristine mires exist basically only in northernmost Lapland and nature reserves [1]. Still there is strong political support for peat mining since peat is a domestic fuel. To preserve existing mires peat mining must be directed to mires that are already drained beyond restoration [2]. To prevent biased politics and conflict of interests an automated method for determining suitable peat mining sites is needed.

Mire naturality index was developed to better describe the condition of mires. The naturality index takes into consideration changes in vegetation and in mire hydrology. The scale is from zero to five, where five indicates totally natural mires. A mire has naturality below two if it has been irreversibly changed by actions of man. In Southern Finland there are only a few mires of state four and five. Autio et al. [3] studied 84 mires of Southern Ostrobothnia. In those 84 mires there were no mires of naturality index five and only two of index four.

The objective of this thesis is to detect ditches and roads from remote sensing data. This is the first phase of a project where the aim is to define mire naturality index automatically. One important descriptor of the state of a mire is the drainage network in mire and its margins. Drainage, even in mire margins, has far-reaching negative consequences to mire hydrology and vegetation and also to aquatic ecosystems connected to the mire [4]. Draining of pristine mires has mostly ended but ditches are still excavated in mire margins [1]. Another important descriptor is the presence of roads near the mire. Thus the method will be tested also for road detection.

Automated detection of structures is done with logistic regression classification together with polynomial modeling of broken segments. Digital terrain model (DTM) created from laser scanning (light detection and ranging, LiDAR) data is used for ditch classification. Most important descriptor of ditches is their height when compared to their

surroundings so they are well seen in DTM. When compared to orthophotography, laser scanning has some great advantages. In orthophotographs tree canopies cause shadows and sometimes block view. Laser waves partly penetrate tree canopies so LiDAR data gives accurate description of ground even in forested areas. Also the quality of LiDAR data is better since the intensity values in orthophotographs might change abruptly due to the changes in image acquisition time and parameters. Road classification is done with grayscale orthophotos and LiDAR height and intensity data.

Logistic regression classification has been successfully used in image classification. Li et al. [5] used multinomial logistic regression and hyperspectral data to classify land cover into 16 classes. Ruusuvuori et al. [6] used logistic regression together with graph cut method [7] to detect spots from simulated images of cell and connection pads from images of printed electronics. Logistic regression was chosen for structure detection since it has proved to be a powerful tool in a variety of image classification tasks and sparse models make it computationally efficient.

One great advantage of logistic regression is the automated feature selection process. The aim was to create a generic feature set that would suit many different classification tasks. Natural scientists can visually identify nature types from remote sensing data. This expertise can be assigned to machine vision tasks if an expert selects ground truth points for training and logistic regression automatically selects best features for the task.

Some of the ditches appear broken after classification due to discontinuities in data and low visibility of lowest ditches. To obtain complete drainage network, broken segments are connected in post-processing step. This is done with polynomial curve modeling presented in [8, 9] and used in cotton fiber [10] and pulp fiber modeling [11]. Based on the results of this thesis, an article was written to a scientific journal in the field of remote sensing [12].

This thesis is divided into eight chapters. Remote sensing basics are presented in chapter two. In chapter three all the features used in classification are introduced. Logistic regression and polynomial modeling are described in detail in chapter four. In chapter five the data acquisition and processing methods are introduced. The actual implementation is shortly described in chapter six and results are presented in chapter seven. Finally, conclusions are discussed in chapter eight.

2. REMOTE SENSING

In this chapter the basics of remote sensing theory are introduced. First Earth ellipsoid, map projections and Finnish coordinate system are described in detail. In second section Earth's surface elevation is discussed. Orthophotography and airborne laser scanning are introduced in next sections. Finally, digital terrain modeling principles are presented.

2.1. Earth ellipsoid and coordinate systems

Earth is an imperfect, flattened ellipsoid which cannot be directly transformed into two-dimensional map. Before any coordinate system can be deployed, we need to accurately model the Earth ellipsoid. Geodetic Reference System (GRS) deployed in 1980, known as GRS80, defines parameters for the reference ellipsoid surface and also for Earth's gravitational field. It is based on the equipotential ellipsoid theory which means that at every point of an ellipsoid surface the gravity potential is equal. The reference ellipsoid is described by four parameters: the equatorial radius of Earth, geocentric gravitational constant, dynamical form factor and angular velocity of Earth. [13]

Based on the GRS80 ellipsoid, coordinate reference systems (CRS) that give locations to geographical entities can be defined. The system used in Finland is ETRS89 (European Terrestrial Reference System 89) which is attached to Eurasian plate. The number 89 refers to the date of the initial definition of the system. This epoch, or reference date, is needed because of the continental drift occurring on Earth. This reference system considers the Eurasian plate as static, so the system will be accurate for only a relatively short time. In literature there is no absolute truth of the speed of continental plates. Zhen [14] estimates the absolute velocity of Eurasian plate to be 0.95 centimeters per year, so in year 2014 the plate has moved nearly 25 centimeters.

The realization of CRS is called coordinate system. It is created by determining control points at specific locations. The coordinates of these points are known, so a certain

location can be fixed to known coordinates.

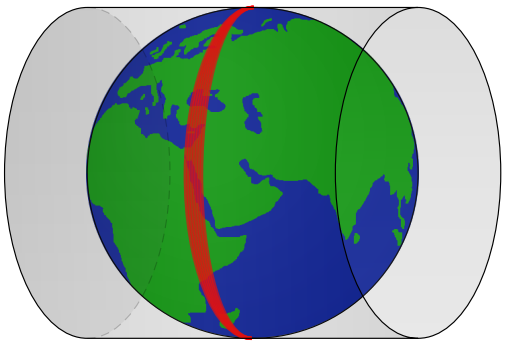


Figure 2.1: TM map projection.

Now to arrive at two-dimensional maps, in addition to coordinate system we need to create a map projection. Map projections are used to mirror Earth's surface to a plane. The projection used in Finnish ETRS-TM35FIN coordinate system is a cylindrical projection where the axis of the cylinder lie in equatorial plane as can be seen in Figure 2.1. This projection is known as Transverse Merchator projection (TM).

Universal Transverse Mercator (UTM) is a cartesian coordinate system used to present whole Earth. It is divided into 60 longitude zones (vertical zones), each zone being 6°wide. UTM uses the TM projection so that the cylinder is placed to each zone separately. Finland situates in zones 34, 35 and 36 but only the zone 35 is used for the projection. It is then widened 5°west and 2°east, totaling 13°and covering the whole of Finland. [15]

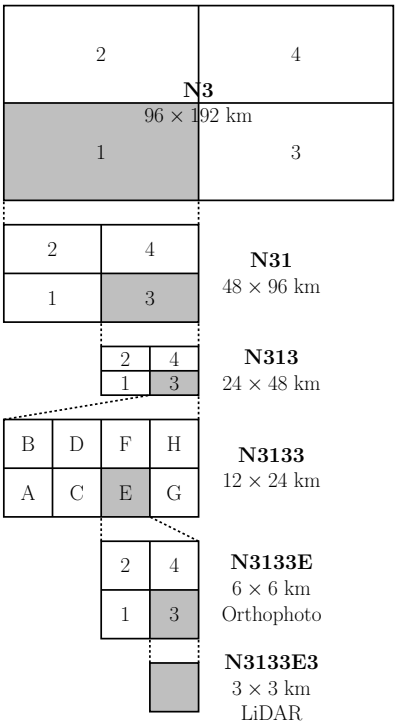


Figure 2.2: Map sheet naming of ETRS-TM35FIN coordinate system.

The data is divided into map sheets with a naming system presented in Figure 2.2. Finland is first divided into 96×192 kilometer blocks which are named with a letter and a number, for example N3 or V5. The letter

presents longitude (from south to north) position and number indicates latitude (from west to east) position. Letters vary from K to X and numbers from 2 to 6. These blocks are further divided into four smaller blocks and a number of smaller block is added to the name. When block size is 12×24 kilometers it is partitioned into eight blocks to obtain square sheets, each of which corresponds to one orthophoto. One orthophoto corresponds to 6×6 kilometer area in nature which in turn consists of four laser scanning data sheets, each representing a 3×3 kilometer area. This systematic naming makes it possible to create an automated process for data download, which will be presented in chapter 5.

2.2. Earth surface elevation

In previous section the reference ellipsoid of the Earth was introduced. The satellite based Global Positioning System (GPS) utilizes this ellipsoid to define horizontal coordinates of a location. However, these coordinates cannot be used in e.g. determining the direction of waterflow. [16] For a more meaningful elevation system, the reference plane used is theoretical ocean surface, geoid. The relation between actual surface, geoid and ellipsoid is presented in Figure 2.3. The letter h presents GPS height, from which the elevation from sea level (H) can be calculated with formula $H = h - N$. Geoid is based on gravitation and rotation as defined in GRS80. Tides and wind are excluded so the geoid presents only the theoretical ocean surface. Oceans are continued to land, thus presenting imaginary ocean surface. [15]

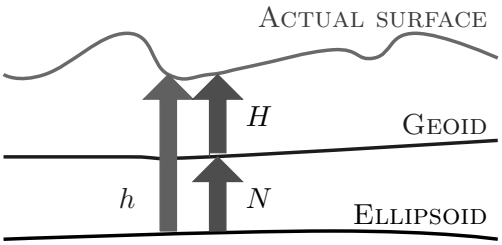


Figure 2.3: Relation between reference surfaces (ellipsoid and geoid) and the actual Earth's surface.

The geoid model is usually defined locally. The Finnish geoid model is called FIN2005 and the height system based on this geoid is N2000 [15]. The error of FIN2005 geoid is less than 5 centimeters in whole of Finland [17].

2.3. Orthophotography

Orthophotos are taken from an airborne platform like a helicopter or a plane. Previously photos were taken with a regular camera to a film, but nowadays digital cameras have very good resolution so they are mainly used in orthophotography. Orthophotos used in this thesis are vertical, which means that the recording camera's axis is perpendicular to the ground. Imagery taken with tilted camera is called oblique. [18, p. 23-24.]

Original aerial photography is often distorted due to e.g. forward motion of the aircraft. Orthophotographs have been orthorectified or, in other words, geometric correction have been applied to them. This is usually done by selecting many ground control points and referencing them to an existing map. [18, p. 41]

One orthophoto provided by National Land Survey of Finland (NLS) is 12000×12000 pixels in size and one pixel corresponds to 0.5×0.5 area in nature. The spectral resolution (number of spectral channels) of these orthophotos is four and the spectral channels are red, green, blue and near infrared.

2.4. Airborne laser scanning

Airborne laser scanning (ALS) is a remote sensing method for ground altimetry measurements, usually referred to as LiDAR. In literature the word LiDAR is considered to be an acronym of Light Detection And Ranging [18, 19]. However, according to Ring [20] and Oxford English Dictionary, LiDAR origins from 1960s as a portmanteau of the words "light" and "radar". Both of these origins indicate the relationship between LiDAR and radar (radio detection and ranging). Radar is a device that transmits radiowaves that bounce of objects and return to a sensor which records the received energy [18, p. 24]. LiDAR has the same principle but since it is a laser, it transmits visible or near-infrared light pulses instead of radiowaves [21]. The time it takes for light to return to sensor is measured and the distance of an object is defined accordingly [18, p. 25]. Most airborne LiDAR sensors can measure multiple returns for one pulse. This is a big advantage when measuring for example forest terrain - even though tree canopy might block part of the pulse, both canopy and ground returns will be detected [18].

Lasers have been used for remote sensing purposes since the early 1960s. In 1962 researches from Massachusetts Institute of Technology (MIT) successfully bounced laser

pulse from the surface of the Moon [20]. During 1980s, as new airborne instruments were developed, LiDAR became a tool for terrain mapping [19]. Before 1995 LiDAR sensors were custom made and expensive, but since 1995 a commercial off-the-shelf instrument market has developed and today LiDAR technology is widely used for topographic mapping [19].

LiDAR data used in the implementation of this thesis is provided by NLS. The measurements have been done from aircraft flying at an altitude of approximately 2000 meters. The laser scanning device is an active sensor so it uses self-generated energy. The point density is in minimum 0.5 points per meter, maximum point distance being around 1.4 meters. Mean error of altitude measurements is at maximum 15 centimeters. Footprint, the size of laser beam on ground, is 50 centimeters and scan angle is +/- 20 degrees. [22]

The data is in a point cloud in laz-format which can be read with e.g. LAStools-program [23]. One point in point cloud has at least the following parameters: X, Y and Z coordinates, time stamp of initial pulse, intensity value, number of the returning pulse (with maximum of three return pulses in total) and classification. The classification is done by NLS first automatically and then interactively with the help of stereo orthophotography. Most common classes are 1 for unclassified points, 2 for ground points, 3 for low vegetation points, 9 for water points and 13 for overlapping points. Low vegetation points come from objects the laser pulse has partly penetrated - from multiple returns they are the returns that are not the last. Ground points present the surface that is the lowest detected from an aircraft. [22] The calculation of height from ground is done from the distance measure, flight path information and calibrated parameters of the laser scanning device [24].

2.5. Digital terrain modeling

Digital terrain modeling is used to obtain a representation of land surface. These models can be calculated from e.g. stereographic orthophotographs, ground survey together with GPS measurements or LiDAR elevation data [25]. In this thesis the LiDAR data is used for digital terrain modeling. It has accurate height information and it is preclassified as described in previous chapter so exact surface presentations are easy to obtain.

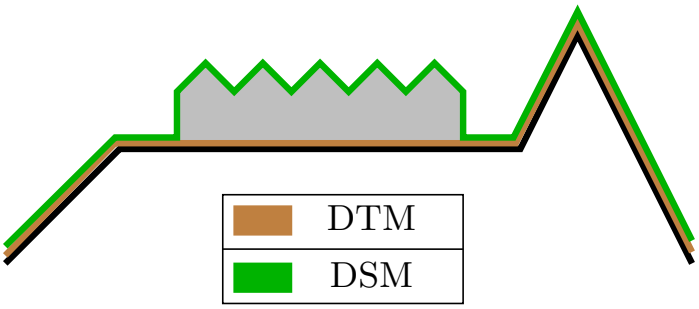


Figure 2.4: Difference between DSM and DTM

The concept of digital terrain modeling includes all land surface models. While DTM is usually considered to be the model of ground, digital surface model (DSM) is a model of ground and all the objects like trees and buildings on it [25]. The difference between DSM and DTM can be seen in Figure 2.4.

Since NLS's LiDAR data is preclassified, it is easy to create both DSM and DTM. For DSM all the points are used, but erroneous points that result from e.g. cloud echoes have to be filtered out. In DTM creation only points preclassified as ground points are used. Digital models can be 3-dimensional representations of a surface, but models used in this thesis are all 2-dimensional raster images. In raster DTM the pixel intensity values are height from sea level.

3. FEATURES

A feature is a numerical presentation of specific image property. Features are obtained with feature extraction, where different operations are applied to the image to calculate an n-dimensional feature vector that represents some image object. In this thesis the aim was to create a large generic feature set so many different objects can be described with a subset of these features. The most descriptive features for presenting specific objects (ditches or roads) are selected from this feature set with automated feature selection presented in next chapter.

In this chapter all the features used are presented. They are roughly divided into classes that best represent the methods with which they are created. Main classes are features obtained by filtering, local statistical properties, features calculated with morphological operations and local binary patterns (LBP). In the end of the chapter, thresholding, wavelets and image gradients are presented, since they do not fall into any of the other categories. All feature names are written in **boldface**.

3.1. Spatial filtering

Spatial filtering in image processing is a technique where a filter (a.k.a. kernel), usually a 2-dimensional matrix, is used to obtain information of pixel surroundings. For each pixel in an image the filter is placed so that the filter center is on the pixel, as in Figure 3.1. The new value for pixel z_5 (filtering response) is obtained be multiplying pixels with corresponding filter coefficients and then summing all values. [26, p. 105]

The process of moving the kernel over the image and taking the sum of products as a new value is called correlation. Convolution is a method closely related to correlation, the only difference is that in convolution the kernel is rotated 180 degrees. [26, p. 150] So with symmetrical kernels convolution and correlation results are the same, but with unsymmetrical kernels they can differ greatly.

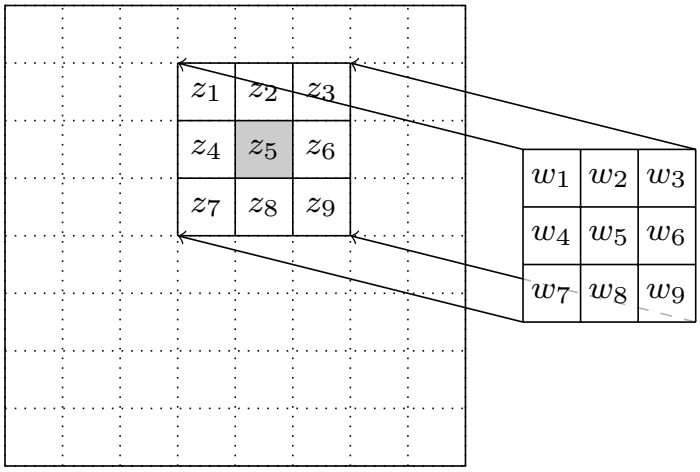


Figure 3.1: Filtering operation for pixel z_5.

Different filters that are used for filtering are presented next. Example filters are mostly 3×3 or 5×5 in size but filters used in features are up to 53×53 in size.

$$F_{edgesH} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \quad F_{edgesV} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

Figure 3.2: Prewitt filters for horizontal (left) and vertical (right) edge detection.

Edge emphasizing filtering is used to highlight linear changes of intensity. Different filters are used for horizontal and vertical edges. Prewitt filters shown in Figure 3.2 are examples of such filters.

Average filters are designed to smooth local regions of an image. The basic average filter of size 5×5 is shown in Figure 3.3.

$$F_{average} = \frac{1}{25} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figure 3.3: 5×5 average filter.

Gaussian lowpass filter (lpf) follows the shape of normal distribution. Gaussian filters emphasize the center pixel and its closest neighbors while giving less weight to pixels further away. Parameters needed to create a Gaussian filter are the size of the filter and standard deviation (std). Two example Gaussian filters are shown in Figure 3.4.

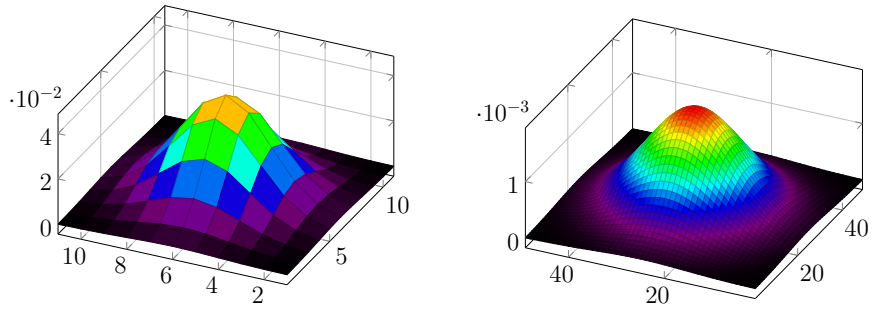


Figure 3.4: Gaussian lowpass filters of size 11×11 (left) and 49×49 (right) with stds of 1.91 and 9.56, respectively.

All filters presented previously are square but sometimes it is convenient to use other shapes. Circular filters cover the pixel neighborhood evenly so the distance from each border pixel to the center is about the same. Two circular structuring elements of different sizes are shown in Figure 3.5.

$$F_{circular} = \begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 0 & 1 & 1 & 1 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline \end{array}, \quad F_{averageC} = \begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 0.077 & 0 & 0 \\ \hline 0 & 0.077 & 0.077 & 0.077 & 0 \\ \hline 0.077 & 0.077 & 0.077 & 0.077 & 0.077 \\ \hline 0 & 0.077 & 0.077 & 0.077 & 0 \\ \hline 0 & 0 & 0.077 & 0 & 0 \\ \hline \end{array}$$

Figure 3.5: Circular structuring element (left) and circular average filter (right) of size 5×5.

Filtered images can be processed further by subtracting filtered image from original image, or by filtering image with two filters of different sizes and subtracting these from one another. **Gaussian filtering difference** is obtained by filtering image with Gaussian lpf and subtracting this image from the original. **Average filtering difference** is done by filtering image with two average filters of different sizes and subtracting the one filtered with smaller filter from the other. **Circular average filtering difference** is otherwise the same as average filtering difference, but the filter is circular. 5 pixels wide example filters are shown in Figure 3.5, where the average filter is on the right.

3.2. Thresholding and h-maxima transformation

Thresholding is a simple operation that results in a binary image. A scalar value is used as a threshold, and pixels that have intensities greater or equal than the threshold are given value 1 and those with smaller intensity values are given value 0. Thresholding is especially efficient in detecting roads from orthophotos since roads (that are not shadowed by trees) appear light in orthophotos.

H-maxima transformation is an image processing technique that suppresses all regional maxima less than threshold h and subtracts h from other values. Regional maxima are connected components of pixels that have constant intensity value and which are surrounded by boundary pixels with lower intensities. For 2-dimensional images connectivity can be either 4 or 8. In 4-connectivity diagonally connected pixels are not included. [27]

Masking with h-maxima transformation is initiated by filtering the image with a small average filter. Then h-maxima transformation is applied to the filtered image. Transformation result is then subtracted from filtered image.

3.3. Morphological transformations

Structuring elements are used in morphological operations to define size and shape of pixel neighborhood used in operations. They are like filters but contain only values 0 and 1, where 1 marks pixels belonging to the neighborhood. For grayscale images morphological erosion is defined as the minimum value in the pixel neighborhood and morphological dilation as the maximum value. Consider we have image I and structuring element B and we mark erosion with $I \ominus B$ and dilation with $I \oplus B$. Morphological opening is erosion followed by dilation [26, p. 668]:

$$I \circ B = (I \ominus B) \oplus B, \tag{3.1}$$

and closing is dilation followed by erosion:

$$I \bullet B = (I \oplus B) \ominus B. \tag{3.2}$$

Now we can define **morphological top-hat transformation** and **morphological bottom-hat transformation**. Top-hat transformation is the subtraction of I and morphologically opened I [26, p. 668]:

$$T_{hat}(I) = I - (I \circ B), \tag{3.3}$$

while bottom-hat transformation is the subtraction of closed I and I:

$$B_{hat}(I) = (I \bullet B) - I. \tag{3.4}$$

(a) Original DTM (b) Top-hat (c) Bottom-hat

Figure 3.6: Morphological top-hat and bottom-hat transformations.

Interpretation of grayscale erosion and dilation is quite simple: erosion emphasizes darker points (lower intensities) while dilation emphasizes brighter points (higher intensities). Defining top-hat and bottom-hat transformations is not as simple, so the results are visualized in Figure 3.6. The transformation was done with 5 pixels wide circular structuring element to a DTM where ditches are clearly visible.

3.4. Local statistical properties

Local statistical properties used in this project include mean, variance, third and fourth image moments, spectral energy, entropy, range and standard deviation. These properties are calculated from square windows of varying sizes. In following equations $\mathbf{z} = z_1, z_2, \ldots, z_{mn}$ is point set included in local window of size $m \times n$.

Mean (μ) is the sum of pixel intensities divided by the number of pixels. **Standard deviation** (σ) and **variance** (σ^2) are closely related to each other. Both can be used to

present image contrast, or in other words the spread of intensity values around the mean [26, p. 97]. Standard deviation is defined as follows:

$$\sigma = \left(\frac{1}{mn} \sum_{i=1}^{mn} (z_i - \mu)^2 \right)^{\frac{1}{2}}, \tag{3.5}$$

and variance is the square of std:

$$\sigma^2 = \frac{1}{mn} \sum_{i=1}^{mn} (z_i - \mu)^2. \tag{3.6}$$

The common operation of std and variance is the sum of difference between mean and pixel values divided by the number of pixels, so the more the pixel intensities vary, the bigger the sum.

Variance is also known as the second image moment. Image moments are defined in [26, p. 97] with probability distributions of pixel intensity levels. However, in this thesis the (normalized) moments are calculated directly from intensities, as was done with variation. So the formula for M^{th} moments is:

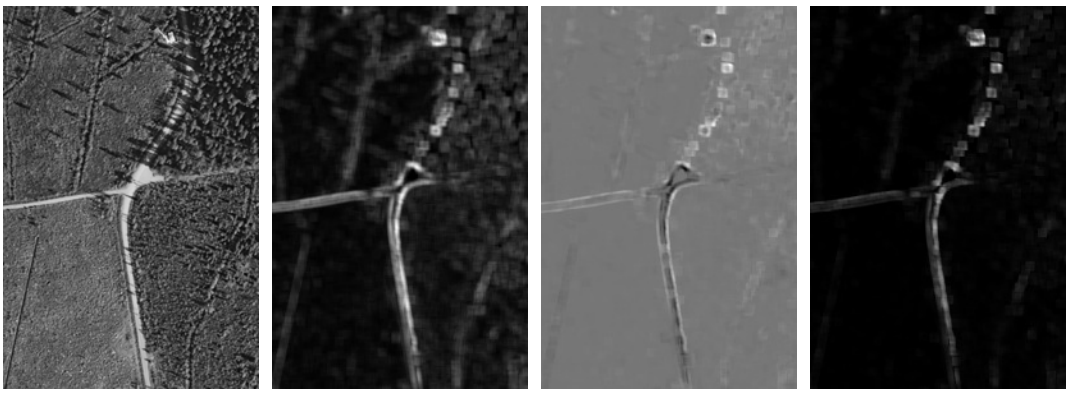
$$\text{mom}_M = \frac{1}{mn} \sum_{i=1}^{mn} (x_i - \mu)^M. \tag{3.7}$$

Third image moment and **fourth image moment** are also used as features. It is difficult to define what these moments actually mean in images so they are visualized in Figure 3.7 together with variance. The operations were applied to RGB orthophoto that was first transformed to grayscale image. Square neighborhood width was 7, so $m \times n = 49$. Variance describes the spread of intensity values so it is especially high on road edges. Since third moment is lifted to the power of three, there are also negative values and this makes it different from second and fourth moments.

Spectral energy density is calculated from Discrete Fourier Transform (DFT). If we have an image $f(x, y)$ of size $M_f \times N_f$, the DFT is calculated with equation [26, p. 235]

$$F(u, v) = \sum_{x=0}^{M_f - 1} \sum_{y=0}^{N_f - 1} f(x, y) e^{-j2\pi(ux/M_f + vy/N_f)}. \tag{3.8}$$

Here the discrete variables u and v are given values $u = 0, 1, 2, \ldots, M_f - 1$ and $v =$



(a) Grayscale orthophoto (b) Second moment (variance) (c) Third moment (d) Fourth moment

Figure 3.7: Image moments calculated from grayscale orthophoto.

$0, 1, 2, \ldots, N_f - 1$.

Each pixel in DFT image $F(u, v)$ has a real (real(u, v)) and imaginary (imag(u, v)) part, which are squared and summed to obtain spectral energy of an image block [26, p. 245]. The magnitude of 2-dimensional DFT, also known as Fourier spectrum, is defined as

$$|F(u, v)| = [\text{real}^2(u, v) + \text{imag}^2(u, v)]^{1/2}, \tag{3.9}$$

and the spectral energy density is obtained with the equation

$$P(u, v) = |F(u, v)|^2 = \text{real}^2(u, v) + \text{imag}^2(u, v). \tag{3.10}$$

Entropy is calculated from histogram H which tells the number of pixels with same intensity values. Number of bins in a histogram of 8-bit grayscale image is 256 so the histogram is defined as $H = h_1, \ldots, h_{256}$. The equation for entropy is [27]:

$$E = \sum_{i=1}^{256} (h_i * \log(h_i)), \tag{3.11}$$

where log is the natural logarithm.

Range is the difference between smallest and largest values of local neighborhood. These values can be obtained with e.g. grayscale erosion and dilation described previously in this chapter.

3.5. Local binary patterns

LBPs are used to describe the texture of pixel surroundings in relation to the pixel itself. They were first introduced by Ojala et al. [28]. The initial approach to local binary patterns is presented in Figure 3.8. Using center pixel g_c as a threshold we assign one to neighboring pixel if it is equal to or bigger than the center pixel and zero otherwise, as in Figure 3.8 (c). These values are then multiplied with two to power of pixel index, zero being the index of first pixel and seven the last. Finally we get the local binary pattern by summing the new values of neighbor pixels, which is done in Figure 3.8 (e). This presentation is basically an 8-bit binary presentation of a number. Example in Figure 3.8 is written 0100 1011 in binary code.

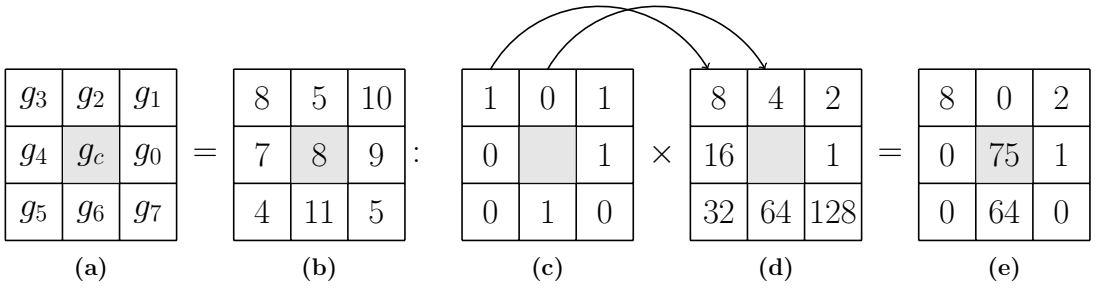


Figure 3.8: Local binary pattern calculation.

The original LBP was calculated with square neighborhood. In generalized approach a radius R and number of samples N are given and LBP is calculated from a circular neighborhood. Given center pixel g_c and surrounding pixels $(g_0, g_1 \dots g_{N-1})$, the formula for general LBP is as follows [29]:

$$\text{LBP}_{N,R} = \sum_{p=0}^{N-1} s(g_p - g_c)2^p, \tag{3.12}$$

where

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

If we have radius R, point index p as used in equation 3.12 and α being angle between two points, we can define x-coordinate of a point with $R\cos(p\alpha)$ and y-coordinate $-R\sin(p\alpha)$, respectively. Weighted average of pixel values is used in cases where four

pixels intersect the sample location. Three examples of generalized approach are shown in Figure 3.9, first one having 8 samples and radius of 1, second one with 12 samples and a radius of 2 and third one with 16 samples and radius of 3.

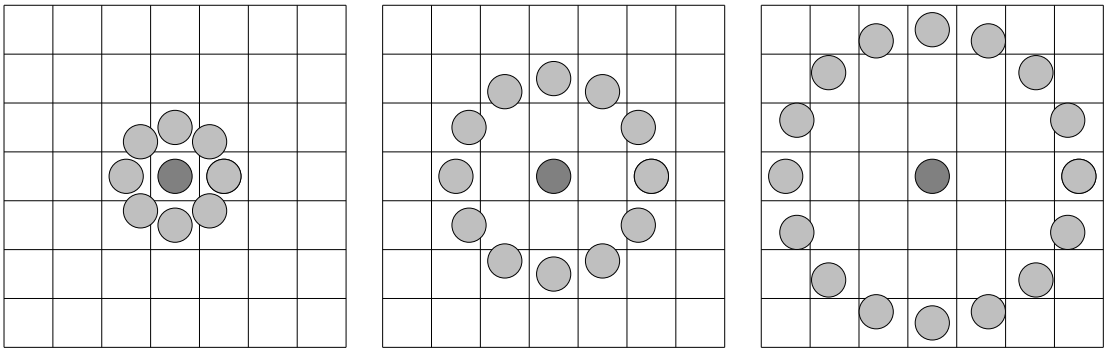


Figure 3.9: Generalized LBPs with different radii and number of sample points.

There are many variations of LBP that are used as features. Many of these modifications aim to make LBP more robust to noise. **Improved Local Binary Patterns** (ILBP) were introduced by Jin et al. [30]. Instead of using the gray value of center pixel as a threshold, ILBP uses mean value of all pixels in a local neighborhood. The center pixel is thresholded with this mean as well. The formula for ILBP is as follows:

$$\text{ILBP}_{N,R} = \sum_{p=0}^{N-1} s(g_p - g_{mean})2^p + s(g_c - g_{mean})2^N, \tag{3.14}$$

where

$$g_{mean} = \frac{1}{N+1}\left(g_c + \sum_{p=0}^{N-1} g_p\right). \tag{3.15}$$

The function $s(x)$ is defined as in equation 3.13. Since the center pixel is calculated to the sum, the maximum value of ILBP is 2^N greater than LBP's maximum value.

Median Binary Patterns (MBP) are closely related to ILBP, but instead of mean value, the median value is used as a threshold. MBP was introduced by Hafiane et al. [31] and the formula is as follows:

$$\text{MBP}_{N,R} = s(g_c - g_{median})2^N + \sum_{p=0}^{N-1} s(g_p - g_{median})2^p, \tag{3.16}$$

where

$$g_{median} = \text{median}(g_0, g_1, \ldots, g_{N-1}, g_c) \tag{3.17}$$

and the function $s(x)$ is defined as in equation 3.13.

In next LBP modifications an additional threshold t is taken into account. In **Local Ternary Patterns** (LTP), as proposed in [32], the difference between neighboring and center pixel is coded with three values:

$$\text{LTP}_{N,R} = \sum_{p=0}^{N-1} s_3(g_p, g_c, t)2^p, \tag{3.18}$$

where

$$s_3(g_p, g_c, t) = \begin{cases} 1, & g_p \geq g_c + t \\ 0, & g_c - t \leq g_p < g_c + t \\ -1, & \text{otherwise.} \end{cases} \tag{3.19}$$

So the scalar t is added to or subtracted from the value of the center pixel and these values are used as threshold. LTP cannot be easily expressed as a binary number since an additional value, -1, is used in coding.

We can extend LTP to **Improved Local Ternary Patterns** (ILTP) as we extended LBP to ILBP, so we use mean value instead of center pixel as a threshold. ILTP was introduced by Nanni et al. [33] and the equation is:

$$\text{ILTP}_{N,R} = s_3(g_c, g_{mean}, t)2^N + \sum_{p=0}^{N-1} s_3(g_p, g_{mean}, t)2^p, \tag{3.20}$$

where s_3 is defined as in equation 3.19 and g_{mean} is calculated with equation 3.15.

Robust Local Binary Patterns (RLBP) are used to improve robustness against small changes in local intensities [34]. We change the LBP equation $(g_p - g_c)$ to $(g_p - g_c - t)$, where t is a threshold as in LTP. This means that 1 is assigned to local neighbor if the neighbor intensity is greater than the sum of center pixel and threshold. The equation for RLBP is as follows:

$$\text{RLBP}_{N,R} = \sum_{p=0}^{N-1} s(g_p - g_c - t)2^p, \tag{3.21}$$

where $s(x)$ is defined as in equation 3.13.

If we consider the task at hand, detecting linear structures that are aligned in different angles, we need to consider rotation invariance. Ojala et al. [29] introduced the concept of rotation invariant LBPs. In Figure 3.10 three cases of rotation invariant patterns are presented. Each line presents rotated patterns that in fact are the same if rotated to same angle. Black dots are ones and white zeros and point indices are as in Figure 3.8(a), so on the left are the rotations with smallest possible value.

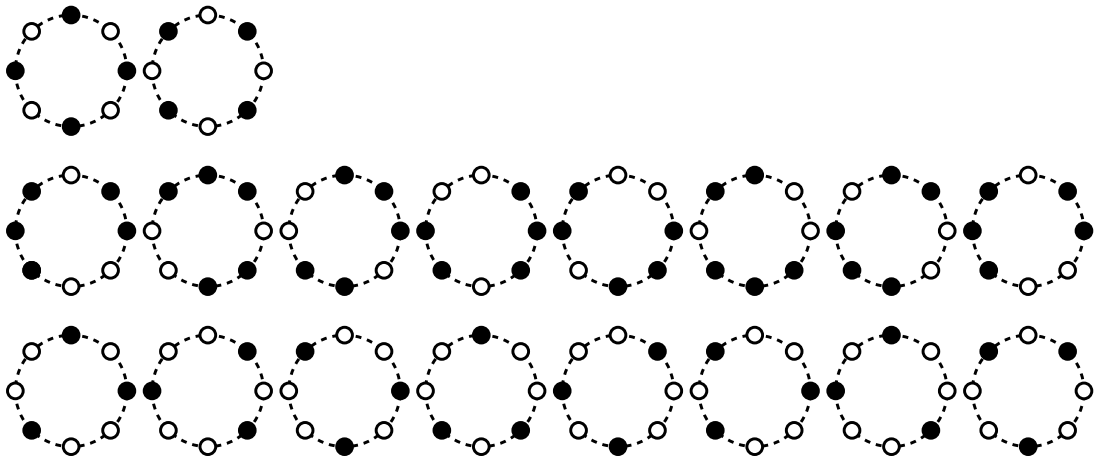


Figure 3.10: Rotation invariance in LBPs, $\bullet = 1$ and $\circ = 0$.

Rotation invariance is used in LBPs where maximum resulting values are two to power of 8, 12 or 16 and minimum is zero. In patterns where also center pixel is thresholded, rotation invariance is applied to pattern before adding the center pixel thresholding result. This excludes patterns with additional threshold (LTP, ILTP and RLBP), so rotation invariance was implemented to LBP, MBP and ILBP.

Local contrast describes the strength of texture and is calculated with circular LBP operator. It is basically the same as variance, but when compared to local variance presented in previous sector, it is calculated from circular area surrounding the pixel. Local contrast is calculated by first subtracting the mean from each LBP point, then lifting the result to power of two and dividing it with number of samples [35]:

$$\mathrm{VAR}_{N,R} = \frac{1}{N}\sum_{p=0}^{N-1}(g_p - \mu)^2, \tag{3.22}$$

where

$$\mu = \frac{1}{N} \sum_{p=0}^{N-1} g_p. \tag{3.23}$$

So the local contrast tells us the sum of differences between the mean and pixel values. By lifting the difference to power of two, negative values are discarded since the sign of difference is insignificant.

Multi-resolution LBP is an extension of LBP where a larger than usual spatial support area is used. This can be done by combining LBP operator with Gaussian filtering. In filtered image, the spatial support area of each sample in LBP neighborhood is as large as the filter. [35]

Gaussian filter bank is a set of Gaussian filters of different sizes and standard deviations that are used in multi-resolution LBP. Scale k is the number of filters in a filter bank and effective area is the area that information is collected from. The radius of this effective area is calculated with equation [35]:

$$r_k = r_{k-1} \left(\frac{2}{1 - \sin(\pi / P_k)} - 1 \right), \quad k \in \{2, \ldots, K\}, \tag{3.24}$$

where P_k is the number of neighborhood samples at scale k. Smallest radius, r_1, is set to 1.5 which is the shortest distance between the center and the border of a 3×3 sized filter. [35]

The radius for LBP operator of each scale is selected so that the effective areas touch each other. Thus the radius of LBP operator at a scale $k (k \geq 2)$ is halfway between the radius of current scale and that of previous scale [35]:

$$R_k = \frac{r_k - r_{k-1}}{2}. \tag{3.25}$$

Gaussian filters are designed in a way that 95 percent of their mass lies within the effective area, so each pixel in filtered image is weighted sum of the whole effective area. The size of Gaussian filter for scale k is obtained by rounding up the LBP radius, multiplying it by two and adding one to the result:

$$w_k = 2 \left\lceil \frac{r_k - r_{k-1}}{2} \right\rceil + 1. \tag{3.26}$$

The width of a two-dimensional Gaussian ($f_G(x, y)$) can be described using std $\sigma(\sigma > 0)$ if it is handled as a zero-mean distribution of two independent variables with equal standard deviations:

$$f_G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{3.27}$$

where x and y are random variables in a Cartesian coordinate system. [35]

The same function can be expressed with polar coordinates (r, θ):

$$f_G(r, \theta) = \frac{1}{2\pi\sigma^2} e^{-\frac{r^2}{2\sigma^2}}. \tag{3.28}$$

Definite integral is used to construct a two-dimensional analogy of one-dimensional Gaussian integral, also known as the Gaussian error function:

$$\begin{aligned} \text{erf}_{\text{2D}}(r, \theta) &= \frac{1}{2\pi\sigma^2} \int_0^{2\pi} \int_0^{r} e^{\frac{t^2}{2\sigma^2}} t \, \mathrm{d}t \, \mathrm{d}\theta \\ &= 1 - e^{\frac{r^2}{2\sigma^2}}, \quad r \geq 0, \end{aligned} \tag{3.29}$$

where t is a dummy integration variable and r is the integration radius. If we consider Equation 3.29 in a statistical point of view, it gives us the probability that the random variables x and y in Equation 3.27 are within the circle of radius r. [35]

The inverse of the Gaussian error function is expressed as

$$\text{erf}_{\text{2D}}^{-1}(P) = \sigma\sqrt{-2\ln(1 - P)}, \quad P \in [0, 1], \tag{3.30}$$

where P is the probability of x and y being within the circle of radius r. So if we want 95% of the mass of filter to lie within the effective area, we must set P to 0.95. The parameter σ is set to 1 to obtain the error function of "normal standard" distribution.

Given the error function, the standard deviation for Gaussian filter is calculated with equation:

$$\sigma_k = \frac{r_k - r_{k-1}}{2 \times \text{erf}_{\text{2D}}^{-1}(0.95)}. \tag{3.31}$$

The resulting Gaussian filters of scales 2, 3 and 4 and their effective areas in LBP are shown in Figure 3.11. The radii of dashed circles are 1.5, 3.35 and 7.53. The widths of Gaussian filters are 3, 7 and 11. Features calculated with Gaussian filter bank are

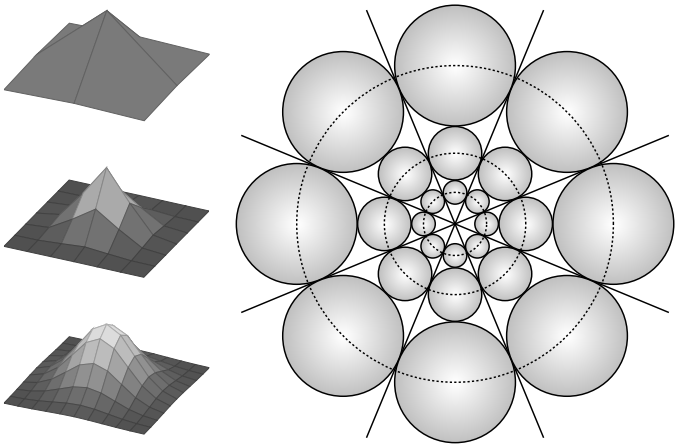


Figure 3.11: Gaussian filters of scale 2, 3 and 4 and their effective areas in 8-bit LBP.

individual LBPs for each scale, the **sum of multi-resolution LBPs** and **Gaussian filter bank difference**, where an image filtered with Gaussian filter of scale $k-1$ is subtracted from image filtered with filter of scale k.

Some LBP operators are combined to get additional features. **Division of LBP and local contrast** is calculated by dividing pointwise LBP image with local contrast image. **Sum of multi-resolution ILBPs** is as sum of multi-resolution LBPs but ILBP is used instead of LBP.

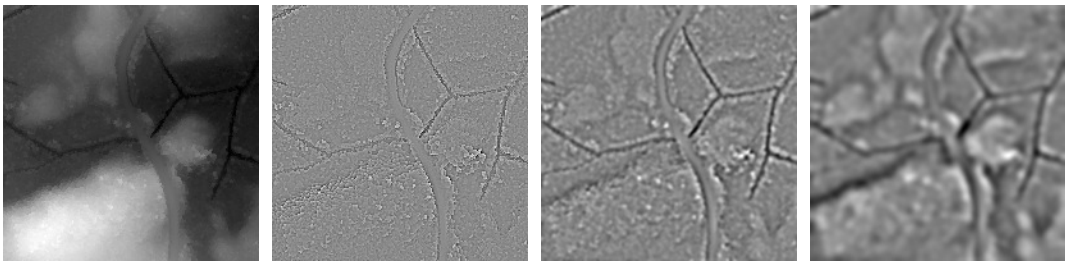
3.6. Wavelet decomposition

Wavelets are based on small waves with varying frequency and limited duration. They fall in the field of multi-resolution theory which analyzes signals or images at more than one resolution. [26, p. 461] Thus the wavelet function used in this thesis creates a 3-dimensional image where each dimension is considered to be one feature, **wavelet decomposition image**. The wavelet transformation is called the à trous (with holes) wavelet transform which, according to Olivo-Marin [36], suits pattern recognition tasks better than original orthogonal wavelet transform.

The wavelets are created by 2-dimensional convolution with 1-dimensional mask. In wavelet decomposition the kernel $[1/16, 1/4, 3/8, 1/4, 1/16]$ is used to first convolve the image row by row and then column by column. Since the kernel is symmetric, the 180 degree rotation has no effect. If we have original image I_0 and convolved image I_1, the

first wavelet plane is found by subtracting the convolved image from the original:

$$W_1 = I_0 - I_1. \tag{3.32}$$



(a) Original DTM (b) First plane (c) Second plane (d) Third plane

Figure 3.12: Three wavelet planes created from DTM

The process is repeated recursively with larger kernel that has been augmented by adding zeros to the kernel. If we have J presenting the total number of wavelet planes and $i \in [1, J]$, the number of zeros added between each number is $2^{i-1} - 1$. From these gaps with zeros comes the name à trous wavelet transform. The formula for recursive wavelet plane calculation is

$$W_i = I_{i-1} - I_i, \quad 0 < i \leq J. \tag{3.33}$$

In Figure 3.12 three first wavelet planes of DTM are shown. First image is the original DTM, following images are wavelet planes 1–3. The effect of augmented kernel can clearly be seen as the spatial resolution diminishes towards higher planes.

3.7. Gradient magnitude

Image gradient is a tool for determining strength and direction of edges in an image. The gradient of image f at location (x, y) is denoted as ∇f and defined as the vector [26, p. 706]

$$\nabla f \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \tag{3.34}$$

This vector points towards the greatest rate of change at specific location in an image. The angle $\alpha(x, y)$ of gradient vector in respect to x-axis can be calculated with equation

$$\alpha(x, y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]. \tag{3.35}$$

The direction of edges is, however, unimportant in detecting ditches and roads since their alignment can be practically anything. More important feature is the strength of edges, or **gradient magnitude** $\text{mag}(\nabla f)$:

$$\text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}. \tag{3.36}$$

As the equation implies, gradient magnitude is actually the length of gradient vector. It determines the value of the rate of change in direction $\alpha(x, y)$. [26, p. 706]

Gradient operators g_x and g_y are partial derivatives at pixel location. The partial first order derivative is defined as follows [26, p. 693]:

$$\frac{\partial f}{\partial x} = f'(x) = f(x + 1) - f(x), \tag{3.37}$$

so it is the digital difference of two consecutive points. When considering two-dimensional image, the partial derivatives are [26, p. 707]:

$$g_x = \frac{\partial f(x, y)}{\partial x} = f(x + 1, y) - f(x, y) \tag{3.38}$$

and

$$g_y = \frac{\partial f(x, y)}{\partial y} = f(x, y + 1) - f(x, y). \tag{3.39}$$

Gradient operators can be calculated with different edge emphasizing kernels, like those presented in Figure 3.2 [26, p. 166]. In this thesis convolution with two different one-dimensional kernels is used for the task. This operation is visualized in Figure 3.13. Darker vectors in Figure 3.13 are used in first convolution to emphasize edges in the image. Then lighter vectors are passed through image to get stronger visibility of continuous edges. The resulting value is placed on z_1. We can express these operations with terms of

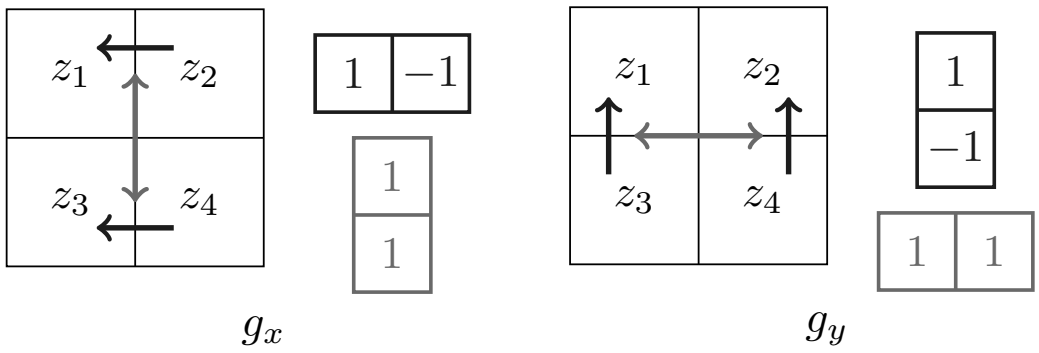


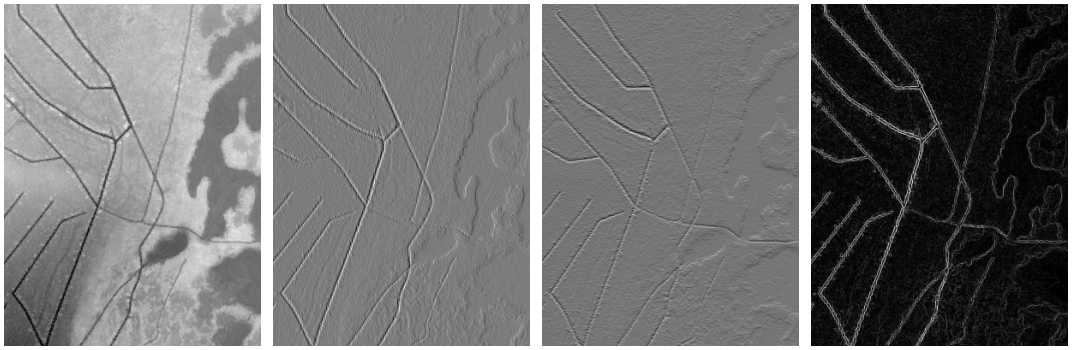
Figure 3.13: Gradient operator acquisition process.

mathematics:

$$g_x = (z_2 - z_1) + (z_4 - z_3) \tag{3.40}$$

$$g_y = (z_3 - z_1) + (z_4 - z_2) \tag{3.41}$$

Since the kernels shown in Figure 3.13 are rotated 180 degrees in convolution, they correspond to filtering with their complements.



(a) (b) (c) (d)

Figure 3.14: Gradient operator images g_x (b) and g_y (c) calculated from DTM (a), and gradient magnitude image (d).

In Figure 3.14 different gradient images calculated from DTM (on the left) are shown. Gradient operator g_x in Figure 3.14 (b) emphasizes especially vertical edges since gradient is perpendicular to the direction of an edge. Thus operator g_y in Figure 3.14 (c) emphasized horizontal edges. Diagonal edges are visible in both images. Figure 3.14 (d) is the gradient magnitude image, representing lengths of gradient vectors.

4. SUPERVISED CLASSIFICATION FRAMEWORK

In this chapter the supervised classification framework is presented. The framework consists of logistic regression classifier together with polynomial modeling as post-processing step.

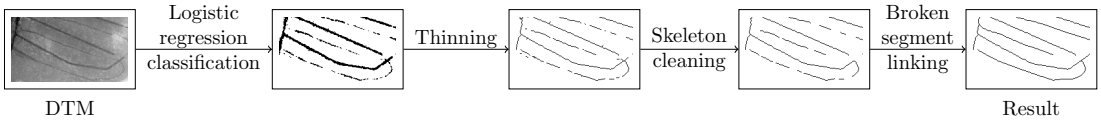


Figure 4.1: Supervised classification framework for ditch classification

Figure 4.1 presents the framework for ditch classification so the example images present ditches. In the classification phase features presented in previous chapter are used as elements for a feature vector $\mathbf{x}$. After the image has been segmented by logistic regression classifier, the result is thinned (skeletonized) and pruned so that polynomial modeling can be successfully applied to image. Road detection framework differs from ditch detection. Ditch classification is done with solely DTM but in road detection DTM, LiDAR intensity images and orthophotos are used for classification. Polynomial modeling is not used in road detection, for reasons that are explained in chapter 7.

The logistic regression classifier with ℓ_1 penalty is presented in section 4.1. Thinning and skeleton pruning process is introduced in section 4.2. Polynomial modeling used for broken segment linking is presented in section 4.3.

4.1. Sparse logistic regression

Sparse logistic regression is a supervised classification method that falls into category of linear classification. Supervised classification means that the classifier is trained with data that has been pre-classified by the user, while unsupervised classifier defines classes automatically without user input. Sparsity describes here the feature selection process where a large set of different features that are given as an input will be diminished to only a

few in the actual model [6]. In linear classification a linear model of image characteristics, or features, is used for the classification. If we have a feature vector $\mathbf{x}$, we can assign class labels with a linear combination of the components of $\mathbf{x}$ with [37, 38]:

$$\text{class} = \begin{cases} 1 & \text{if } \beta_0 + \boldsymbol{\beta}^T \mathbf{x} < 0 \\ 2 & \text{otherwise,} \end{cases} \tag{4.1}$$

where $\boldsymbol{\beta}$ is a weight vector and β_0 is a bias. These are the model parameters of logistic regression classifier.

Logistic regression classifier is based on the *logistic function*:

$$P(t) = \frac{1}{1 + \mathrm{e}^{-t}}. \tag{4.2}$$

This function is plotted in Figure 4.2. Consider we have a feature vector $\mathbf{x} \in \mathbb{R}^n$ presenting n features of pixel i. Logistic regression classifier uses the logistic function to map the linear combination of this feature vector into an estimate of the probability that pixel i belongs to a certain class. It is thus a probabilistic classifier. In Figure 4.2 $P(t)$ describes this probability.

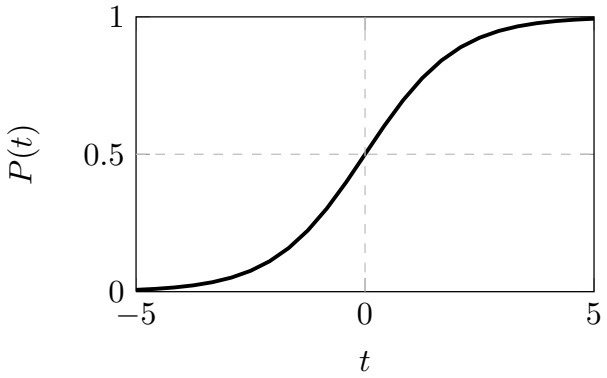


Figure 4.2: Logistic function.

Logistic regression classifier can be used to classify image into two (binomial classifier) or more (multinomial classifier) classes. In this thesis only binomial classifier is used since ditches and roads are classified separately. Thus the classes are ditch and no ditch, and road and no road. These will be later referred to as foreground (ditch or road) and background.

The probabilities of pixel i belonging to foreground ($p(c_F|\mathbf{x})$) and background ($p(c_B|\mathbf{x})$) can be determined by combining equations 4.1 and 4.2. We can define a linear predictor through linear function of the predictors [39]:

$$\begin{aligned} p(c_F|\mathbf{x}) &= \tfrac{1}{1+\mathrm{e}^{-(\beta_0+\mathbf{x}^T\beta)}}, \\ p(c_B|\mathbf{x}) &= \tfrac{1}{1+\mathrm{e}^{(\beta_0+\mathbf{x}^T\beta)}} = 1 - p(c_F|\mathbf{x}). \end{aligned} \tag{4.3}$$

So with negative predictor value as given in equation 4.1 for class 1, the probability is between 0 and 0.5.

In Figure 4.3 the logistic function is visualized. First the linear predictor is defined by multiplying each feature with corresponding element of model $\boldsymbol{\beta}$ which can be considered as a weight of the feature. Model coefficient β_0 is added to the sum of predictors, which is then passed to logistic function to make a prediction.

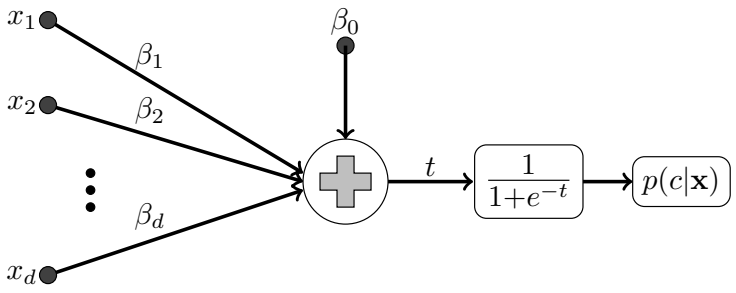


Figure 4.3: Logistic regression classifier

The feature set consists of hundreds of features, some of witch are highly correlated and some do not contain any significant information. We need a regression method that selects only few features that give desirable classification result. Regularization is one way to approach the problem of feature selection. In regularization, some additional information like a penalty is used to arrive to a solution. The regularization method in logistic regression is an extension of LASSO, which stands for Least Absolute Shrinkage and Selection Operator and was developed by Tibshirani [40] in 1996 for linear regression. LASSO is based on least squares approach, but adds a penalty factor to the method. Least squares method minimizes the sum of square error (residual) of the model [26, p. 888].

Given original image y and classification result $\hat{y}$, the residual R is obtained by sub-

tracting the latter from the former:

$$R = y - \hat{y}. \tag{4.4}$$

Now the sum to be minimized is the sum of squares of residual terms:

$$S = \sum_{i=1}^{n} R_i^2 = \|y - \hat{y}\|^2. \tag{4.5}$$

LASSO combines the least squares method with ℓ_1 penalty which can be expressed with equation [39]

$$\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|. \tag{4.6}$$

The theory behind LASSO according to is as follows. Consider we have data $(\mathbf{x}^i, y_i), i = 1, 2, \ldots, N$ where N is the number of initial features. Our input is $\mathbf{x}^i = (x_{i1}, x_{i2}, \ldots, x_{iM})^T$ which presents our M selected coordinates and y_i are their responses. We presume that responses y_i are linearly independent given the input coordinates x_{ij}. The predictors we want to derive are β_0 and $\boldsymbol{\beta} = \beta_1, \beta_2, \ldots, \beta_M$.

We calculate the residual term by combining equation 4.4 and the linear combination of components of $\mathbf{x}$ given in equation 4.1:

$$R_i = y_i - \beta_0 - \sum_{j=1}^{M} \beta_j x_{ij}. \tag{4.7}$$

Now we can determine the LASSO equation [40]:

$$\underset{(\beta_0,\beta)}{\arg\min} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{M} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{M} |\beta_j| \right\}. \tag{4.8}$$

In other words we want to find the $\boldsymbol{\beta}$ with which we get the minimum sum of square error. Parameter λ in Equation 4.8 is our constraining factor since it limits the sum of absolute values of $\boldsymbol{\beta}$ (ℓ_1 penalty). First sum in equations presents the least square error of the model.

Friedman et al. [39] used logistic link function, or logit, to extend LASSO to create the logistic regression classifier. If we consider equation 4.3, the logistic link function is

as follows [39]:

$$\log \frac{p(c_F|\mathbf{x})}{p(c_B|\mathbf{x})} = \log \frac{p(c_F|\mathbf{x})}{1 - p(c_F|\mathbf{x})} = \beta_0 + \mathbf{x}^T \boldsymbol{\beta}, \tag{4.9}$$

where $\log$ is the natural logarithm. Now we can estimate β by maximizing ℓ_1-penalized log likelihood [6, 39]:

$$\left[\sum_{\mathbf{x} \in F} \log p(c_F|\mathbf{x}) + \sum_{\mathbf{x} \in B} \log(1 - p(c_F|\mathbf{x})) - \lambda \|\boldsymbol{\beta}\|_1 \right], \tag{4.10}$$

where F and B are the training sets of foreground and background. Parameter $\lambda > 0$ limits the length of $\boldsymbol{\beta}$ so it affects the sparsity of the result. It is selected by cross-validation, where the training coordinates are divided into two sets. One set is used for creating the model and model accuracy is tested with the other set. In Figure 4.4 an example plot of validation process is shown, obtained from ditch detection model creation. Vertical axis presents the error calculated with cross-validation and horizontal axis presents the value of ℓ_1 penalty, as given in equation 4.6. The error is calculated way beyond the visible x axis but for illustration purposes the plot is cut at 4000. The classifier selected $\boldsymbol{\beta}$ from the point marked with dashed line. The minimum error value 0.01331 was obtained when the penalty had value 5278, but since we have the limiting factor λ, it was not approved.

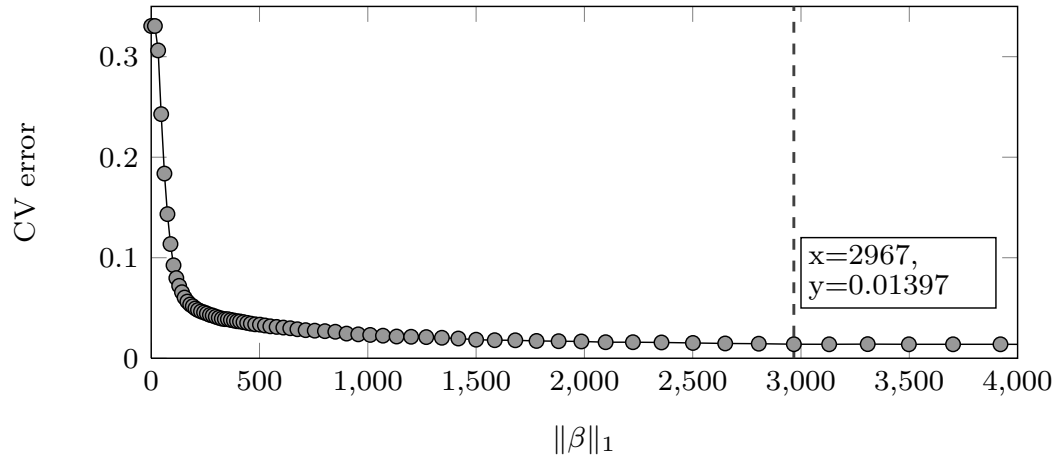


Figure 4.4: Cross-validation error in relation to ℓ_1 penalty of $\boldsymbol{\beta}$.

The model parameters β_0 and $\boldsymbol{\beta}$ are estimated with `glmnet` algorithm created by Friedman et al. [41]. It is a cyclical coordinate descent method which estimates the model coefficients with different values of λ.

The probability images are transformed into binary images presenting ditches or roads. This is done by selecting a suitable threshold between values 0 and 1 to obtain optimal classification. Foreground class is the class with low probabilities, so pixels below the given threshold are those presenting roads and ditches.

4.2. Thinning and skeleton pruning

Thinning is done to obtain one pixel wide skeleton that can be further processed with polynomial modeling. Pruning is the process of removing branches shorter than a threshold. End points are points that are connected to only one other point in a skeleton. Junction points can be considered as crossroads since they are connected to at least three other points.

After classification the resulting binary image is thinned with an algorithm proposed by Guo et al. [42] (Algorithm 1). It is a two-subiteration thinning algorithm that works in a 3×3 neighborhood shown in Figure 4.5.

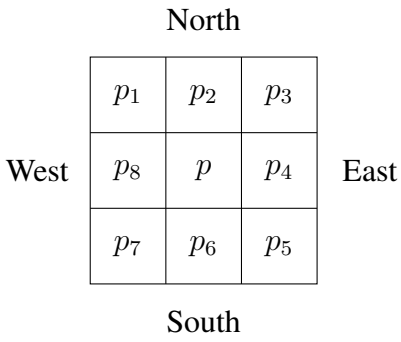


Figure 4.5: Eight-connected neighborhood of pixel p.

As can be seen in Figure 4.5, p_2, p_4, p_6 and p_8 are p's side neighbors, while p_1, p_3, p_5 and p_7 are diagonal neighbors. In following text, we will call pixels having value 1 true and pixels having value 0 are called false. Obviously pixel p is true since the operation is applied to it. Some variables and concepts are introduced next so that the actual algorithm can be presented. Symbols $\wedge$, $\vee$ and $\neg$ are logical conjunction (and), disjunction (or) and negation (not). $C(p)$ is the number of distinct eight-connected components of ones in the neighborhood, which can be evaluated with equation:

$$C(p) = \neg p_2 \wedge (p_3 \vee p_4) + \neg p_4 \wedge (p_5 \vee p_6) + \neg p_6 \wedge (p_7 \vee p_8) + \neg p_8 \wedge (p_1 \vee p_2). \quad (4.11)$$

$C(p)$ can have values from 0 to 4. The equation means that for each false side pixel we check next two pixels clockwise. If one or both of these are true, 1 is added to $C(p)$, so largest possible value is obtained only if diagonal pixels are true and side pixels false.

The connectivity of p is described with variable $N(p)$:

$$N(p) = \min(N_1(p), N_2(p)), \tag{4.12}$$

where

$$N_1(p) = (p_1 \vee p_2) + (p_3 \vee p_4) + (p_5 \vee p_6) + (p_7 \vee p_8) \tag{4.13}$$

and

$$N_2(p) = (p_2 \vee p_3) + (p_4 \vee p_5) + (p_6 \vee p_7) + (p_8 \vee p_1). \tag{4.14}$$

$N_1(p)$ and $N_2(p)$ group neighbors into four connected pairs, where each pair is given value 1 if one or both pixels are true. $N(p)$ is useful in endpoint detection, but in addition it helps create thinner results. [42]

Pixel p is set to false if the following conditions are satisfied:

a. $C(p) = 1$

b. $2 \leq N(p) \leq 3$

c. Apply one of the following:

1. $(p_2 \vee p_3 \vee \neg p_5) \wedge p_4 = 0$ in odd iterations

2. $(p_6 \vee p_7 \vee \neg p_1) \wedge p_8 = 0$ in even iterations.

Condition **a.** is necessary to preserve local connectivity of a neighborhood, so p cannot be deleted if it is in the middle of a curve. Condition **b.** ensures that no end points are deleted. Possible combinations that satisfy condition **c.** are shown in Figure 4.6, where (a) and (b) present odd iterations and (c) and (d) even iterations. Points marked with x can be either true or false. Condition **c.1** removes pixels from north and east boundaries while condition **c.2** removes south and west boundary pixels.[42]

Since the segmented lines have often rugged edges, the skeleton has many excess branches that are removed with skeleton pruning. Branches are lines starting from a junction point and ending in an endpoint that consist of less than N_p pixels. Junction points

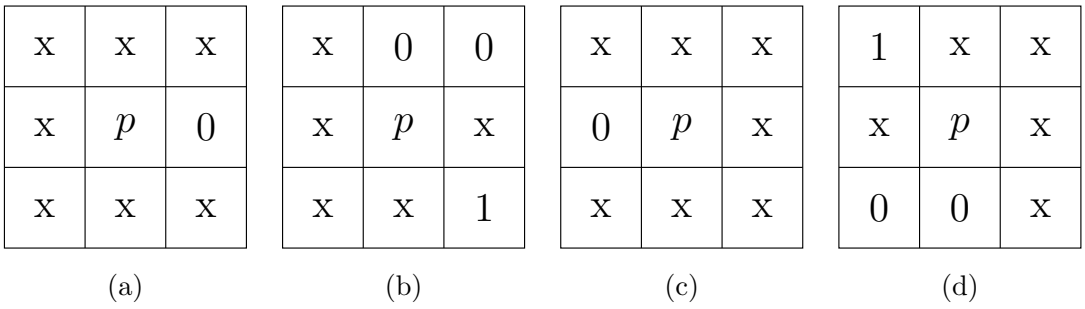


Figure 4.6: Pixel values satisfying conditions **c.1** ((a) and (b)) and **c.2** ((c) and (d)) of Guo's thinning algorithm.

are points from which lines continue to three directions. Branch removal is done with an algorithm proposed by Niemistö et al. [43].

If we have an input image I, temporal image J and a point amount threshold N_p, we create pruned image with following steps [43]:

1) Delete each 3×3 neighborhood of a junction point from I and store the result in J.

2) Delete all connected components from J that do not contain an end point of a connected component in I. Store the result in J.

3) Delete the pixels in I that correspond to connected components in J smaller or equal to threshold N_p and store the result in I.

4) Delete all end points in 3×3 neighborhood of junction points from I and store the result in I.

Connected components are all eight-connected, so diagonal connections count. The last step removes remaining spurs of one pixel in size, but might sometimes create new ones. Thus the algorithm is repeated iteratively until there is no change between original and resulting images. [43]

Sometimes after thinning and pruning small rings remain in the skeleton which can be seen in Figures 4.7(b) and 4.7(c). The process of removing these rings from image I goes as follows:

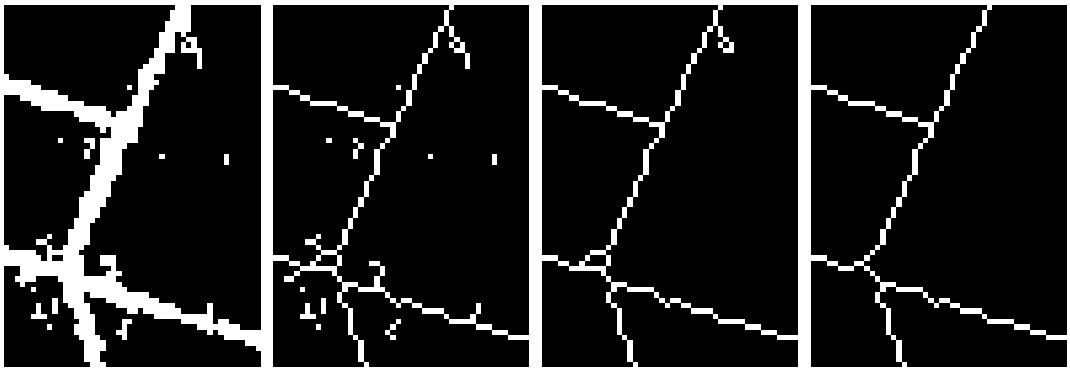
1) Fill holes in the image and save the result in J_1.

2) Erode J_1 with a circular structuring element of radius 6 and save the result in J_2.

3) Set pixels that have value 1 in J_1 and 0 in J_2 as 1 in I.

4) Thin I and remove branches.

Middle steps are applied so that ditches that connect to each other will not be lost, since they might form big rings that are filled in the first step.



(a) (b) (c) (d)

Figure 4.7: Ditch detection result (a) and the same result after thinning (b), pruning (c) and ring removal (d).

Skeletonization results are shown in Figure 4.7. In Figure 4.7(b) is the result of thinning of ditch detection result shown in Figure 4.7(a). Figure 4.7(c) is pruned skeleton and Figure 4.7(d) presents the final result after ring removal.

4.3. Polynomial modeling

After thinning and skeleton pruning, polynomial modeling can be applied to the image. Polynomial modeling is a method where broken skeleton segments are connected to form continuous lines. We repeat the following method for each end point:

1. Beginning from endpoint X_1 find continuous line of length l.
2. Find polynomial model with smallest mean square error for the line.
3. Extend the line and find points P near the extension.
4. If not enough points are found, find nearest endpoint and get a new point set by following a continuous line from that endpoint. If distance is very small, concatenate fibers and continue to step one.

5. Calculate new model for X and P to fill gaps between points.

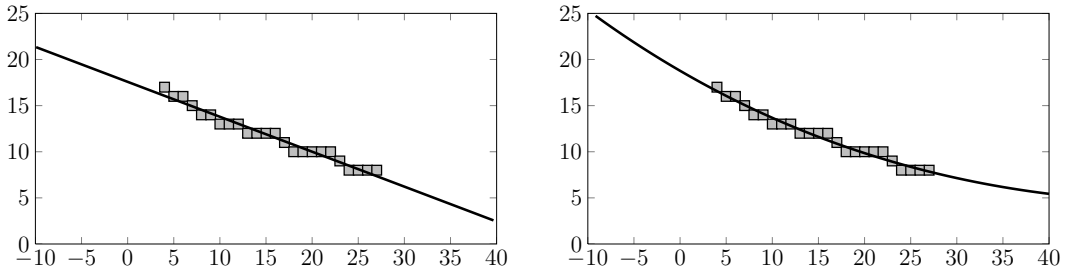
6. If mean square error of resulting model and existing segments is smaller than a threshold, concatenate fibers.

The polynomial modeling principles are as follows. If we have a line segment presenting N coordinates, $X = (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \in \mathbb{R}^2$, the linear model of M^{th} order for the line is $X = H\theta + E$. The model parameters, as given in [11], are

$$H = \begin{pmatrix} 1 & t_1^1 & t_1^2 & \cdots & t_1^M \\ 1 & t_2^1 & t_2^2 & \cdots & t_2^M \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_N^1 & t_N^2 & \cdots & t_N^M \end{pmatrix}, \quad \theta = \begin{pmatrix} \theta_{0,1} & \theta_{0,2} \\ \theta_{1,1} & \theta_{1,2} \\ \vdots & \vdots \\ \theta_{M,1} & \theta_{M,2} \end{pmatrix}$$

and $E \in \mathbb{R}^N$ is a residual term that the model does not explain. The second column in matrix H is vector $\mathbf{t} \in [0, 1]$.

In Figure 4.8 two modeling results are shown. Squares present the original pixels and black lines are modeled lines. Figure 4.8(a) presents first order model of the line, so the new line is straight. In Figure 4.8(b) a second order model of the line is presented.



(a) (b)

Figure 4.8: First (a) and second (b) order models of a line.

There are three different cases for modeling the line: modeling a segment, extending line segment and filling a gap in line. In the first case, $\mathbf{t} \in [0, 1]$ is the same length as X. For each point in X a value is assigned to vector $\mathbf{t}$ by traversing the line and calculating distance to previous point. The equation for the method is [8]:

$$t_i = t_{i-1} + \|X_i - X_{i-1}\| \frac{t_{max}}{l}, \quad i = 1, 2, \ldots, N \tag{4.15}$$

where l is length of the segment, $t_1 = 0$ and $t_{max} = N + 1$. Vector $\mathbf{t}$ is then normalized to arrive at $t_1 = 0$ and $t_N = 1$. In line extension and gap filling a model for existing segment is first calculated. Then model parameter θ is calculated for the existing line, as presented later in this chapter. Now $\mathbf{t}$ is continued with values greater than 1. Suitable values are estimated by finding optimal distance to previous point. In gap filling $\mathbf{t}$ is assigned values from 0 to 1, but vector length is the length of X plus length of the gap. The model parameter θ is calculated with standard least squares method:

$$\hat{\theta} = (H^T H)^{-1} H^T X. \tag{4.16}$$

With $\hat{\theta}$ known, we can get new line from equation

$$\hat{X} = H\hat{\theta}. \tag{4.17}$$

In some cases the line curvature changes towards the end of the line. Polynomial model for changing curvature can be found with simple extension of least squares method. First we calculate a diagonal matrix W with equation $w_{k,k} = (1 - \min(t_k, 1 - t_k))$. For stronger endpoint curvature, $w_{k,k}$ is lifted to power of $j \in [1, 10]$. Now with weighted least squares we get equation [11]:

$$\hat{\theta} = (H^T W H)^{-1} H^T W X. \tag{4.18}$$

The modeling is done two times for the image. In first step only the smallest isolated pixel groups are removed before modeling so ditches that are not well classified and consist of many small pixel groups can be connected. The second time also bigger pixel groups are removed to get a clean result.

Finally the resulting skeleton is dilated with a 3x3 circular structuring element. The testing coordinates are not always in the center of the ditch so dilation is needed to get accurate validation.

5. DATA

Remote sensing data used in this thesis consists of LiDAR point clouds and orthophotographs. In this chapter data licence and different mires used for training and testing are presented. Also the data acquisition process and LiDAR intensity data preprocessing are described.

5.1. Data licence

All remote sensing data used in this thesis is obtained from National Land Survey of Finland (NLS). LiDAR data and orthophotos are under National Land Survey open data licence (Avoimen tietoaineiston lisenssi 2012). The licence gives user rights to freely use and modify data in both commercial and non-commercial purposes. Licence can be found at `http://www.maanmittauslaitos.fi/en/NLS_open_data_licence_version1_20120501`.

5.2. Study sites

The selection of study sites was done with four criteria in mind:

1. Since NLS's LiDAR data does not yet completely cover Finland, main criterion was that the necessary data exists.

2. The study site is presented in the Mire study of Southern Ostrobothnia [3] which is used as a reference guide.

3. In ditch detection the mire site needs to be considerably changed by actions of man so a large drainage network is present.

4. In road detection, naturally, the presence of roads is required.

Ditch training set consists of Lammineva in Kurikka and Pohjaisneva in Alavus. Kalistanneva and its surroundings in Kurikka is used for testing. Road training set includes

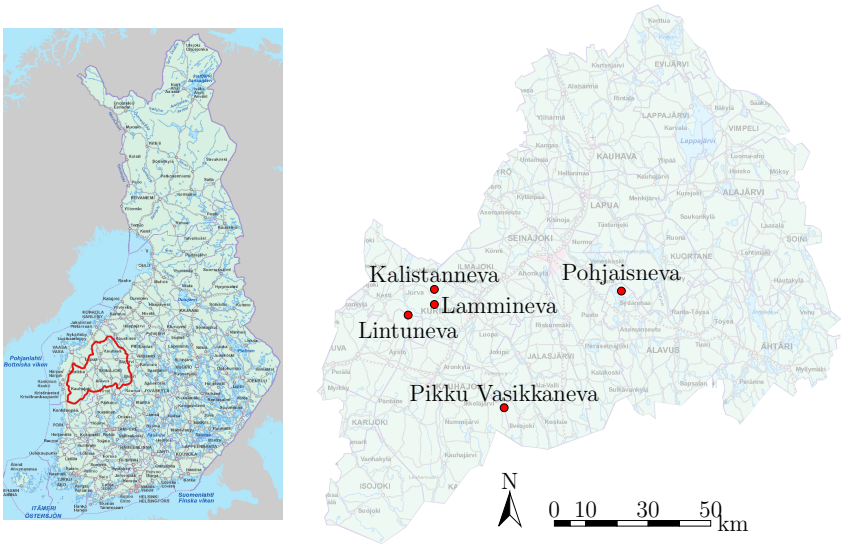


Figure 5.1: Southern Ostrobothnia map and the mires used in this thesis. Contains data from the NLS Municipal Division Database 01/2014.

mires Pikku Vasikkaneva in Jalasjärvi, Pohjaisneva in Alavus and Kalistanneva in Kurikka while test mire is Lintuneva in Kurikka and Teuva. The mire locations are shown in Figure 5.1.

5.3. Data acquisition

All the data is downloaded from kartat.kapsi.fi server where it has been mirrored from NLS's own server. The reason for this is that data from NLS's server has to be searched and downloaded manually. From kapsi server the data can be automatically downloaded via the `wget` command in Unix. The data is divided into packages that are named with ETRS-TM35FIN coordinate system, as presented in Figure 2.2. These names can be calculated automatically from coordinates, so the input parameters for data acquisition function are coordinates of a mire and desired image size in meters. From this information the image block names that are needed for whole image are calculated and, if the images do not exist on computer, first searched and then downloaded from kapsi server. The orthophotos are resized to half the original size so one pixel corresponds to 1×1 meter area in nature. This is done to simplify data fusion, since the DTM pixel size is 1×1 meters. Using smaller pixel size would not be sensible since point distance in LiDAR data is 1.4 meters.

Raster digital terrain model (DTM) is created from LiDAR data by simply fitting the scattered data into a grid. Nearest neighbor interpolation is used to fill empty grid cells. Only points pre-classified as ground points are used. Pixel values in raster image are height values in N2000 height system. An intensity image is also created the same way, with the difference that LiDAR intensity values are used instead of height values. In addition, points classified as low vegetation (trees) are used to create a vegetation image, where treeless areas are given value zero. DSM is created by combining DTM and vegetation image, so that zero values in vegetation image are substituted with corresponding values from DTM. Finally images are cropped and combined to create mire-centered images of desired size.

5.4. Intensity data filtering

LiDAR intensity data is quite noisy so filtering operation is applied to it before training and classification. Often mean filtering is used in noise removal, but in the case of road borders, mean filter would take also other than road pixels into consideration. This would make road border intensities differ from road center intensities [44]. To preserve road edges, *selective filtering* is used to smooth the intensity data. It is a method where different filters are used and one of them is selected based on some criterion [45]. The selective filtering used in this thesis is based on Hachimura-Kuwahara filtering, where 9 different filters are used. These filters are shown in Figure 5.2.

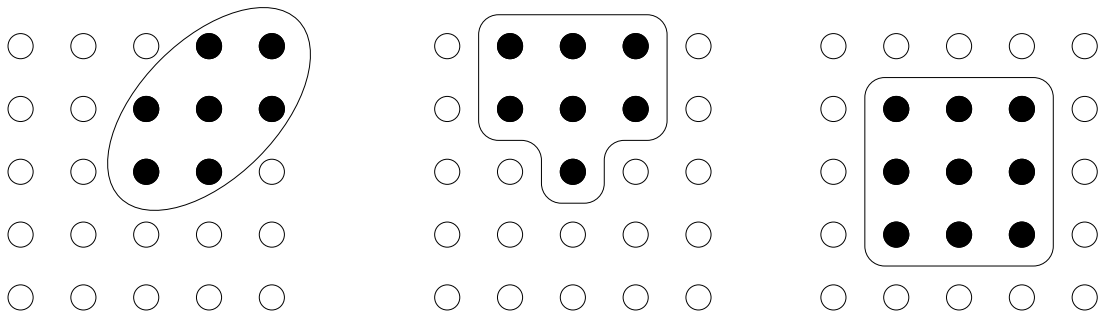


Figure 5.2: Hachimura-Kuwahara filters. First two filters are rotated 90, 180 and 270 degrees to obtain rest of the filters, so there are 9 filters in total.

First two filters in Figure 5.2 are rotated 90, 180 and 270 degrees to obtain filters covering all corners and sides. The last filter is a basic 3×3 mean filter that works well for other than road border points.

In Hachimura-Kuwahara filtering the variance is used as selection criteria. It is calculated from the area that the filter covers, and filter resulting in minimum variance is used to calculate mean that will be the new value for the center pixel [45, pp. 105-107]. Algorithm 1 presents the Hachimura-Kuwahara filtering algorithm. Filtering is done to each pixel separately. For each filter, the variance of pixels inside filter is calculated and if it is smaller than smallest variance thus far, mean of those pixels is saved to a variable $Mean_k$. After all filters are passed through pixel, $Mean_k$ is the mean of filter with smallest variance and original pixel value is replaced with this mean value.

Algorithm 1 Algorithm for Hachimura-Kuwahara filter.

Hachimura-Kuwahara filter

Inputs: $NumberOfRows \times NumberOfColumns$ `image`
`Set of subwindows` $\{W_1, W_2, \ldots, W_M\}$
Output: $NumberOfRows \times NumberOfColumns$ `image`

for $i = 1$ `to` $NumberOfRows$ **do**
  for $j = 1$ `to` $NumberOfColumns$ **do**
    `let` $SmallestVar = LargestPossibleValue$
    for $k = 1$ `to` M **do**
      `place the subwindow` W_k `at` (i, j)
      `let` $Var_k =$ `the variance of the elements inside` W_k
      if $Var_k < SmallestVar$ **then**
        `let` $SmallestVar = Var_k$
        `let` $Mean_k =$ `the mean of the elements inside` W_k
        `let` $OutputCandidate = Mean_k$
      end
      `let` $Output(i,j) = OutputCandidate$
    end
  end
end

Based on Hachimura-Kuwahara filter, Xudong et al. [46] developed the flatness mean filter (FMF). In FMF, the variance is calculated with height values since roads are uniform in flatness but are often higher than their surroundings. Then the intensity data is used to calculate mean values, and mean intensity corresponding to smallest variance is used as new intensity.

6. IMPLEMENTATION

In this chapter the implementation of the work is presented. All coding was done with Matlab®. Some functions were pre-given. These include the logistic regression classifier [47], original LBP code, filtering operations, wavelet and h-maxima transformations and skeletonization and skeleton pruning functions. Thus, functions programmed by the author include rest of the features, data acquisition and DTM creation, graphical user interface for coordinate selection, flatness mean filtering function, polynomial modeling function (excluding skeletonization) and performance evaluation calculations.

The training and test points were selected manually. A possibility of extracting points from topographic maps by selecting objects of certain color with thresholding was considered, but this proved to be more unreliable method than manual selection. LiDAR data and orthophotos are relatively new when compared to topographic maps, so new structures do not exist in maps and some of the older ditches had been dried. Also the topographic maps are not always totally accurate when compared to e.g. LiDAR data and manual checking of correct locations was as time consuming as the manual selection of coordinate points. In the training phase it is very important that the points are actual ditch or road points and not just near a ditch or a road. To ease the coordinate selection process, a graphical user interface (GUI), shown in Figure 6.1, was created for the task. With the interface it is possible to define classes and select pixels or areas from image belonging to a certain class. Also the removal of isolated pixels or pixels in a selected area is possible.

Training was done with two images (and two coordinate sets) in ditch detection and three images in road detection. It was noted that one image is not enough for a reliable model. In ditch detection two images were enough for successful classification. Road detection proved to be a bit harder a task so three training images were used. The minimum number of training images that gave a satisfactory model were used since coordinate selection is a slow process and it must be done for test image also.

Even though feature set can be large and selection is automated, some rough initial

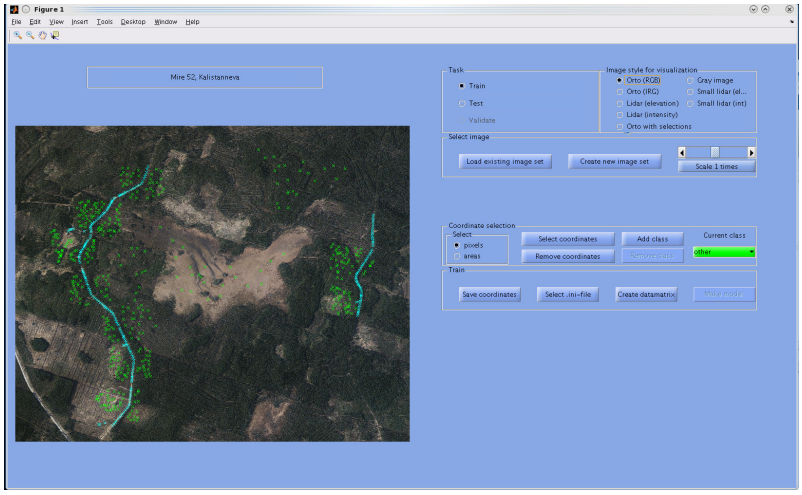


Figure 6.1: Road point selections in GUI.

assumptions were done to reduce computational burden of the training phase. Since the most descriptive property of a drainage network is its height in relation to surroundings, only raster DTM was used for ditch detection. The true height of a ditch changes as ground elevation changes, so features that use only the value of one pixel or mean of local neighborhood were not used. The features used in ditch detection are shown in Appendix 1. Parameter range is given in format [min:step:max] and features are given in the order they appear in Chapter 3. Some features take multiple parameters as inputs, but not all parameter values are necessarily used together. For example in LBPs the radius of one is only used with 8 points and a radius greater than 10 is used for 12 and 16 points. In ditch detection there were 1065 features in total in the training phase.

In road detection LiDAR intensity images, orthophotos, DTM and DSM were used. RGB components of orthophotos were combined to create a grayscale image. LiDAR intensity images were filtered with flatness mean filter, as described in chapter 5.4. Also, since there are no trees on a road, DSM created with ground and low vegetation points was used. Here also the unprocessed intensity values mattered, so e.g. basic thresholding was used. Since multiple images were used for classification, it was not possible to use all the features with all images due to excessive computational burden. Features for road detection are shown in Appendix 2. The images from which features were calculated are given abbreviations, some of which were invented solely for the feature table. Since they are not commonly used, they were not included in abbreviations listing. There were 1468 features in total in road detection in the training phase.

7. EXPERIMENTAL RESULTS

The proposed method was tested with two different linear structures - ditches and roads. For ditch detection only a raster digital terrain model is used for classification. In road detection also orthophotographs, DTM and other LiDAR data were used.

In this chapter performance evaluation metrics are introduced, followed by the results of ditch and road detection.

7.1. Performance evaluation

Before calculating performance evaluation metrics, we need to first calculate hits and misses of testing coordinates. There are four possible outcomes for each coordinate point. These outcomes can be presented in a confusion matrix shown in Figure 7.1.

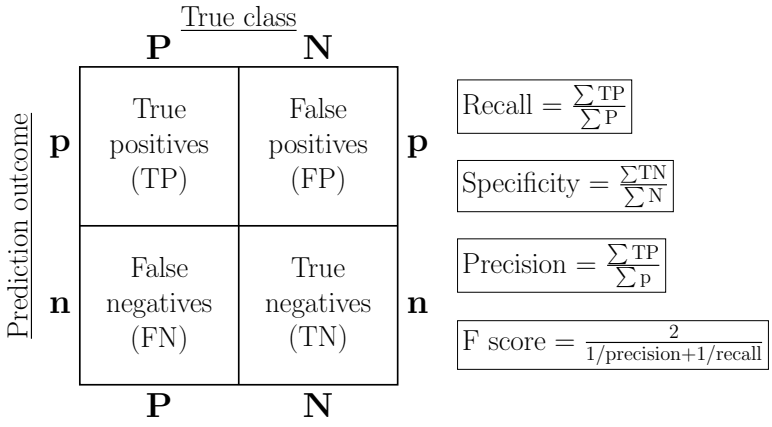


Figure 7.1: Confusion matrix for performance evaluation. [48]

When the classifier makes a correct guess, we call it a hit. When the guess is wrong, we call it a miss. True positives (TP) are hits when the true class is positive, while false negatives (FN) are misses. When true class is negative, false positives (FP) are misses and true negatives (TN) are hits.

In following equations, based on article by Fawcett [48], the sum ($\sum$) means the number of points in a class. Recall (a.k.a. true positive rate, sensitivity) is the percentage of

correctly classified positives from all positives (P):

$$\text{Recall} = \frac{\sum \text{TP}}{\sum \text{P}} = \frac{\sum \text{TP}}{\sum \text{TP} + \sum \text{FN}}. \tag{7.1}$$

Specificity (a.k.a. true negative rate) in turn is the percentage of correctly classified negatives from all negatives (N):

$$\text{Specificity} = \frac{\sum \text{TN}}{\sum \text{N}} = \frac{\sum \text{TN}}{\sum \text{TN} + \sum \text{FP}}. \tag{7.2}$$

Precision (a.k.a. positive predictive value) is the percentage of true positives from all points classified as positives (p):

$$\text{precision} = \frac{\sum \text{TP}}{\sum \text{p}} = \frac{\sum \text{TP}}{\sum \text{TP} + \sum \text{FP}}. \tag{7.3}$$

F score is the harmonic mean of precision and recall:

$$\text{F score} = \frac{2}{1/\text{precision} + 1/\text{recall}} = \frac{2 \times \sum \text{TP}}{\sum \text{p} + \sum \text{P}} = \frac{2 \times \sum \text{TP}}{2 \times \sum \text{TP} + \sum \text{FP} + \sum \text{FN}}. \tag{7.4}$$

Recall and specificity describe how well positives and negatives are detected. Precision grows when number of false positives diminishes, so it tells us how precise the classifier is. F score improves when precision and recall improve.

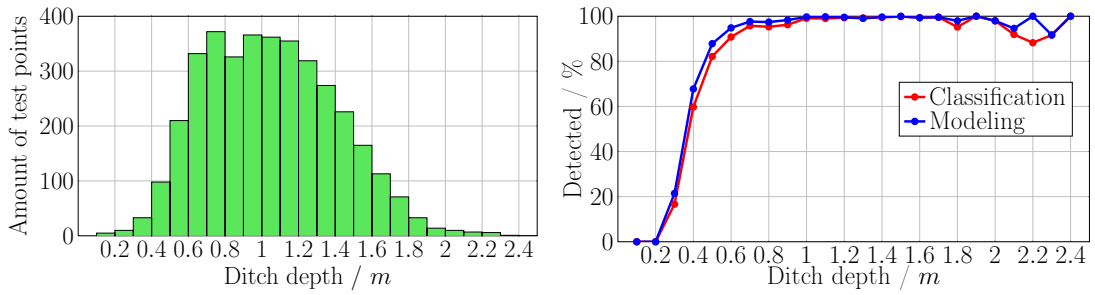
7.2. Ditch detection results

Ditch detection was done solely from DTM. Probability image resulting from ditch classification was transformed into binary image with threshold 0.5. The threshold was selected since it is in the center of the logistic function shown in Figure 4.2 and visual assessment supports this conclusion. Polynomial modeling parameters were defined with careful testing and visual evaluation of testing results. It was quite difficult to find optimal length of gaps to fill and best mean square error (MSE) threshold since too big values cause erroneous linking and too small values prevent linking of many broken segments. Maximum gap length was chosen to be 35 and MSE was 0.3 when candidate line was chosen by nearest end point and 0.4 otherwise. These parameters were chosen through experiment-

ing. The number of testing points used for evaluation metrics was 5802, from which 3732 were ditch points.

There were 1065 features in total from which 120 features were selected for the model. Most used features were LTP, ILTP and rotation invariant ILBP, but also most of the other features were used at least once. This does not directly tell us the importance ranking of the features since the number of parameters for features varies. Also since the value range of features varies, the importance of features cannot be determined from the model coefficients β. However, since most of the features were used in the model, it can be said that the feature set was suitable for ditch detection task.

Coordinates were divided into classes of equal size according to their depth. Depth is calculated by maximum difference in 9x9 meter area around the pixel. In Figure 7.2(a) the histogram of testing point depth is presented. Results of classification are shown in Figure 7.2(b). In y-axis is the percentage of correctly classified points, x-axis presents ditch depth. The lower percentage in deepest ditches is due to the small amount of points in those classes since one undetected point can lower greatly the total detection percentage. From Figure 7.2(b) can be seen that ditches deeper than 1 meter are well detected. Lower ditches, however, often break due to low visibility in LiDAR data, but their detection was improved with polynomial modeling. This suggests that polynomial modeling makes the method more robust.



(a) (b)

Figure 7.2: Histogram of ditch depth in testing point locations (a) and percentage of ditches found in each depth class before (red) and after (blue) post-processing.

One example area of results is shown in Figure 7.3. Figure 7.3(a) shows the orthophoto of the study site and Figure 7.3(b) is the DTM of selected area. In Figure 7.3(c) the unprocessed classification result is shown. Figure 7.3(d) and Figure 7.3(e) are results of two runs of polynomial modeling. Colors are used to represent different methods of

the process: red is the original classification result, orange presents linking with nearest points, yellow lines are linked with nearest endpoint and white lines are linked ditches that were very close to each other.

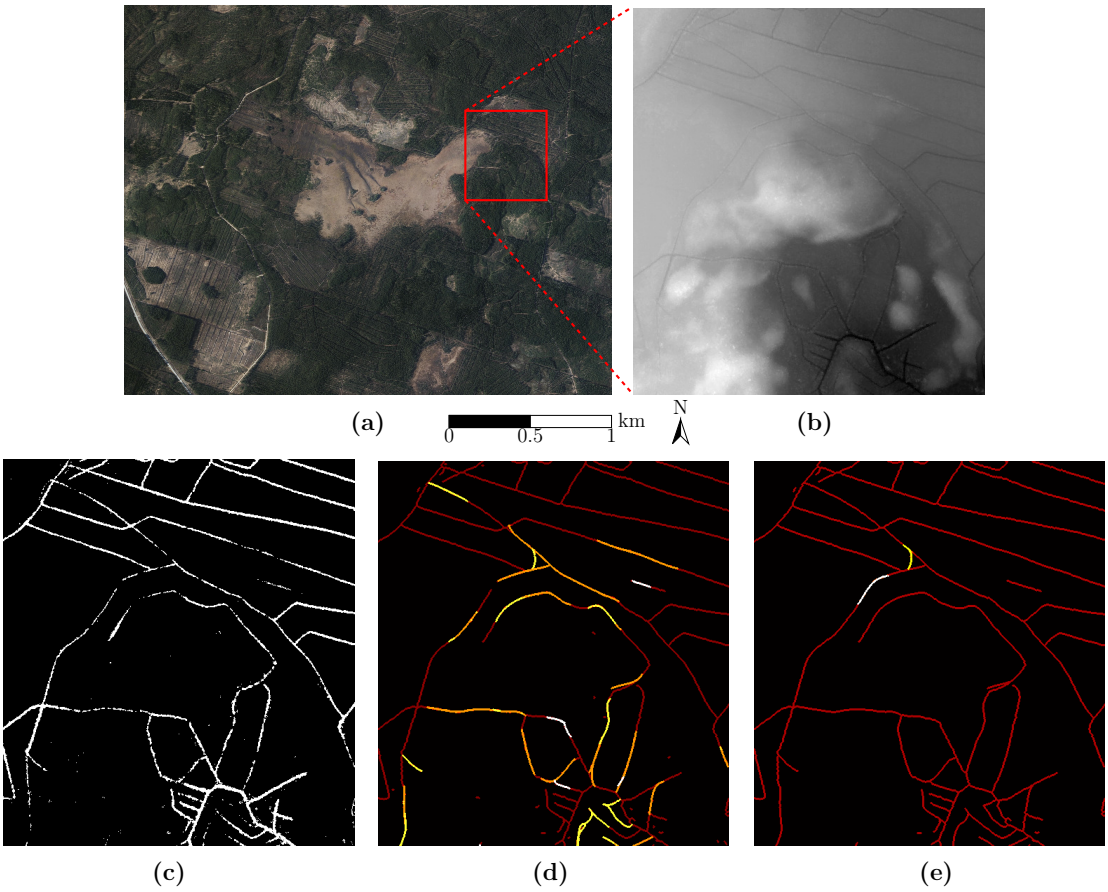


Figure 7.3: Orthophoto of Kalistanneva (a), DTM (b), original classification result (c) and results of first (d) and second (e) run of polynomial modeling. Contains data from the NLS Laser Scanning Database 03/2012 and Orthophoto Database 08/2013.

Judging by visual assessment the locations of detected ditches correspond to actual ditch locations and polynomial modeling improved the results. Even curved lines were linked successfully as can be seen in Figure 7.3(d).

The performance of our classifier was evaluated quantitatively with precision, recall, specificity and F score. Values were calculated before and after polynomial modeling. Results can be seen in Table 7.1. Recall improved with polynomial modeling which means that the number of correctly classified ditch points increased. The number of false positives slightly increased causing the decrease of specificity and precision. F score, which is a harmonic mean of recall and precision, also increased, which tells us that the

increase of recall outweighs the decrease of precision.

Table 7.1: Evaluation metrics for ditch detection.

	TP	FN	TN	FP	Recall	Specificity	Precision	F score
Classification	3378	354	2065	5	0.9051	0.9976	0.9985	0.9495
Modeling	3630	102	2064	6	0.9727	0.9971	0.9984	0.9853

Testing points for ditch detection were selected so that there are points on most of the ditches so the recall value is quite accurate. The number of negative points, however, is too small to make definite conclusions from evaluation metrics but by visual assessment it can be seen that there are not many false positives.

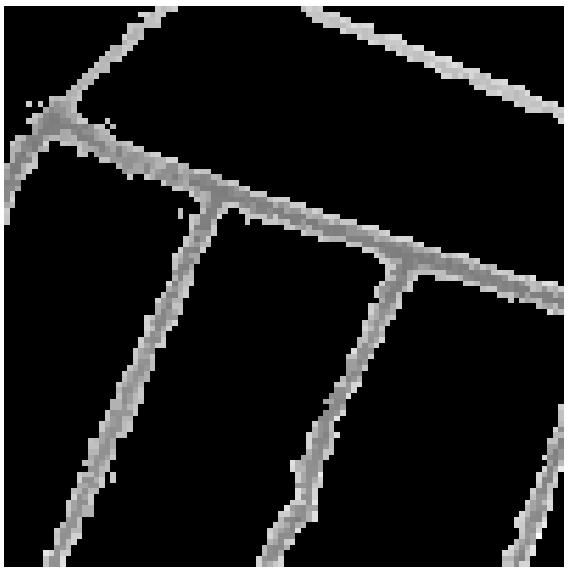


Figure 7.4: DTM multiplied with unprocessed classification result.

The unprocessed segmentation result reaches coordinates outside ditches as can be seen in Figure 7.4, which was obtained by multiplying the DTM with the segmentation result. There are two possible reasons for this. Firstly, there are not many negative training points near ditch borders. Secondly, some of the training coordinates have been placed just outside ditch. Since the ditches at their narrowest are just one or two pixels wide, it is quite understandable that some of the manually selected coordinate points have been misplaced. However, to further improve the results, more precision is needed in training coordinate selection process.

The results are very encouraging. Recall value of 0.97 is a good result in image classification. With further improvements of the method it is possible to raise the recall value

even higher. Detection of ditches deeper than half a meter was already very successful so improvements should be directed especially on detection of lower ditches.

7.3. Road detection results

In logistic regression model for road detection there were 123 features in total. Most used features were edge emphasizing filtering, LTP and ILTP, rotation invariant ILBP, multi-resolution LBP, different local statistical properties and morphological transformations.

Probability threshold was selected by calculating probability images of one of the three training mires, then thresholding this image with values between 0.01 and 0.95 with step size of 0.01. These features were used as training set for logistic regression classifier. However, the actual model was not used but instead biggest value of model β was detected and the corresponding threshold was selected for the task. This threshold was 0.25.

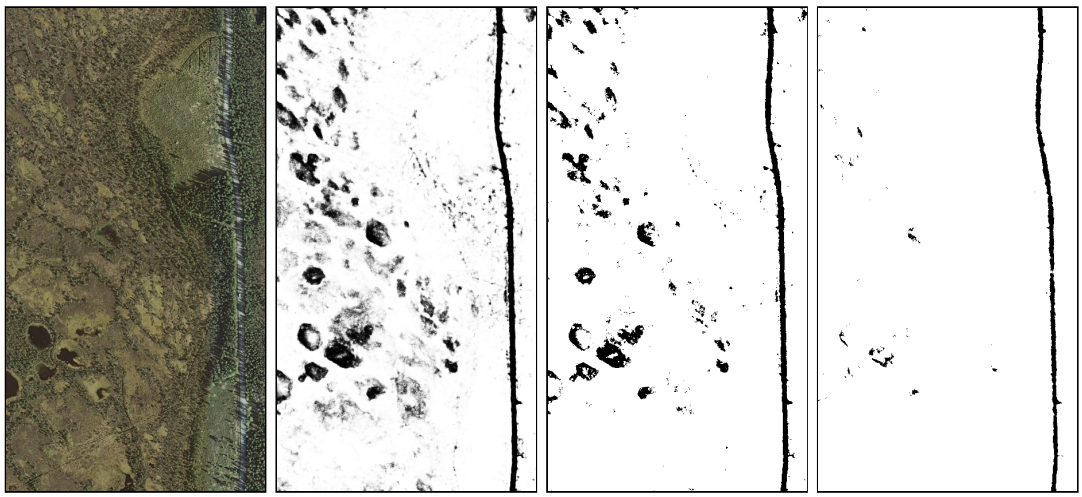


Figure 7.5: Orthophoto of Lintuneva (a), classification result (b), binary image obtained with threshold 0.25 (c) and 0.01 (d). Contains data from the NLS Laser Scanning Database 03/2012 and Orthophoto Database 06/2012.

In Figure 7.5 an example of road detection result is shown. Here the road class has value 0 and background class value 1, while in ditch detection class values were the other way around. The image is from the center of Lintuneva where a road intersects the mire, so it is an interesting test case for the mire naturality project. Image on the left is the original orthophoto and probability image is shown in Figure 7.5(b). The last two images are thresholded results, where (c) was thresholded with value 0.25 and (d) with value 0.01.

There are some circular structures in a mire that are false positives. With a threshold low enough these structures can be mostly removed as can be seen in Figure 7.5(d), but this also removes many road segments. Even though the road in Figure 7.5 stayes intact with lower threshold, gaps in some other roads became too long to connect with polynomial modeling. This means that the classifier has not been taught to exclude these structures. Thus, we need either a bigger training set or some additional information to reliably detect all the roads and only the roads. On the positive side, the wide asphalt road in Figure 7.5 was very well detected even though tree canopies cause shadows in orthophotographs. These shadows are the main reason for using also LiDAR data, otherwise roads could be detected from solely orthophotos with thresholding and filtering.

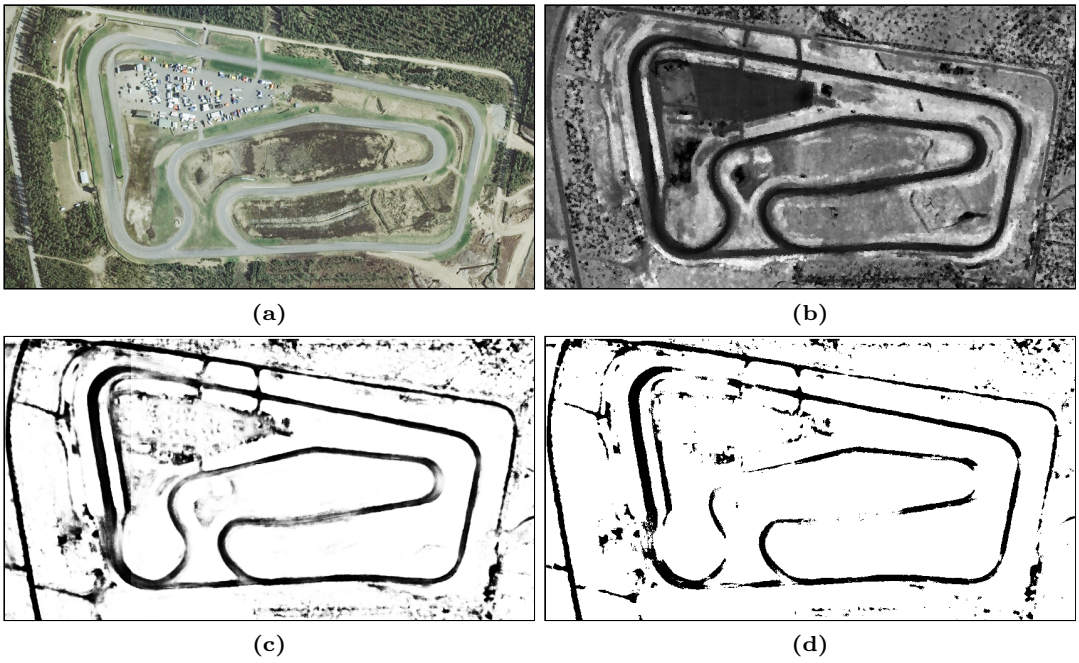


Figure 7.6: Orthophoto (a) of a race-course, LiDAR intensity image (b), probability image (c) and result thresholded with value 0.25 (d). Contains data from the NLS Laser Scanning Database 03/2012 and Orthophoto Database 06/2012.

In Figure 7.6 another classification result is presented. First image is the orthophoto of area where a race-course is present. Second image is the LiDAR intensity image filtered with FMF. In Figure 7.6(c) is the probability image and in (d) is binary image obtained with threshold 0.25.

The horizontal road on top of the image is gravel road, while the race-course is asphalt. Figure 7.6(d) tells us that both types of road can be detected by the classifier. Also the

parking lot was not classified as a road, which means that the shape of the road is a significant cue in addition to intensity values. Some parts of the race-course are not as well detected, so with a threshold of 0.25 they disappear.

Evaluation metrics for road detection are presented in Table 7.2. They are calculated from binary images obtained with different thresholds. There were 10041 testing points in total from which 1128 were road points and 8913 background points.

Table 7.2: Evaluation metrics for road detection.

Threshold	TP	FN	TN	FP	Recall	Specificity	Precision	F score
0.5	1032	96	8416	497	0.915	0.944	0.675	0.777
0.25	991	137	8608	305	0.879	0.966	0.765	0.818
0.01	721	407	8878	35	0.64	0.996	0.954	0.765

Evaluation metrics show that the results are as goos as in ditch detection. With low threshold value (0.01) precision improved but recall decreased, so many false positives were removed but also many of true road points were deleted. With large threshold value (0.5) there were many false positives but also roads were detected well. The best F score was obtained with threshold of 0.25. Thus, this threshold is a compromise between high recall value with the expense of precision and high precision value with the expense of recall.

The polynomial modeling function was tested for roads, but it did not improve the results. This is due to the fact that there are so many false positives in the classification result. Broken segments can be connected but this results in increased number of false positives since existing false positives might be connected also. To remove false positives from the skeleton pruning is needed, but when the length of branches to remove is big enough to remove false positives, also some road segments are removed. Thus if the number of false positives is decreased, so is the number of true positives. With low threshold value many false positives can be removed, but some gaps in roads become very long. If the maximum gap length in polynomial modeling is set very high, it will result in many wrong connections. We can draw a conclusion that the classification with the current set of features does not provide a satisfactory basis for polynomial modeling.

Road detection results could be improved by dividing the problem into smaller parts. Here all roads were in same class, but they could be divided into gravel and asphalt road classes. Multinomial logistic regression would then be used for classification since the

number of classes is greater than two. This idea could be taken even further by dividing the road points into road center and road border points. Now the classifier tries to bundle different types of roads into one class which makes the problem complicated. With more classes the description of one class would be more simple and probably more accurate. Also more information would be obtained with single classification.

8. CONCLUSIONS

In this thesis ditch and road detection from remote sensing data with logistic regression classifier was presented. Ditch classification was done with only DTM, and polynomial modeling was applied to classified results to link broken segments. The detection results were very accurate for ditches deeper than half a meter and below that the ditch detection accuracy decreased as ditches became lower. Broken segments were successfully connected with polynomial modeling and this improved the detection of low ditches, too. Roads were detected from different images with logistic regression, but polynomial modeling was not applied. The road detection accuracy does not correspond to values obtained from ditch detection so the method should be further improved for it to function reliably. Yet success in ditch detection indicates that logistic regression is a suitable method for this application.

Logistic regression can create computationally efficient models that can classify images very accurately. However, the data used in this thesis is very heterogeneous and there is noise and variation in measurements. For example, orthophotographs collected from different study sites might look very different since the intensity values are not consistent. Thus, the models might be complicated and results are not always as accurate as desired, as was noted with road detection. Variations in data also make it necessary to test the method thoroughly in the future. The results apply to a very limited material since only one test site was used for both ditch and road detection, so more data is needed to evaluate the reliability and generality of the method.

Some improvements could be achieved by adding new features to the feature set since suitable features are the foundation of successful classification. New features are easy to add due to the modularity of the implementation. Also combining existing features could help, for example all features could be multiplied with one another. A bigger training set with more study sites could improve the results, too, and make the method more general. It is important to give the classifier a comprehensive training set of background points

so it knows what kind of features to exclude. These improvements would increase the computational burden in the training phase so the use of external computing resources is recommended. However, by improving the training process the number of features in the model would decrease and the classification phase would be faster and more reliable.

One possible application for logistic regression classification of remote sensing images could be automated creation of topographic maps. The method should be deployed first in a semi-automated manner, where the resulting map would be checked by human before it is accepted. Currently topographic maps of NLS are created manually from aerial photography. Classifier could be taught to recognize buildings, fields, forests and other units appearing in maps. Ditches and roads are also important parts of these maps, so the road detection method should be further improved. The method for ditch detection should be further tested to get a better idea of method's reliability. Now it has been tested only in forested areas but testing in urban and rural areas would be required too. The results of ditch detection imply that this kind of application is achievable with the classifier, but it would need a comprehensive training set and comprehensive evaluation.

Ditches and roads in mire and its margins are important descriptors of the natural state of a mire. With the method for ditch and road detection presented in this thesis, the automated analysis of the natural state is one step closer. Ditch information will be utilized by determining the percentage of drained mire margins and giving additional weight to ditches inside a mire. Also drainage network location is essential knowledge when modeling waterflow patterns in and around a mire. Roads that cross mires affect negatively in mire naturality, so they must be taken into consideration when determining the naturality index. The next step in mire naturality analysis will be mire type classification, since the effects of drainage vary according to mire type. This will be done by combining logistic regression classification of surface texture and ground height gradient to determine whether the mire is convex or concave.

REFERENCES

[1] Lindholm, T. and Heikkilä, R. Destruction of mires in Finland. In: Lindholm, T. and Heikkilä, R., (eds.), Finland - land of mires, volume 23 of The Finnish environment. Finnish Environment Institute, 2006. pp. 179–192.

[2] Ylönen, M. and Simola, H. The Finnish peat mining paradox: political support to environmental calamity. In: Lindholm, T. and Heikkilä, R., (eds.), Mires from pole to pole, volume 38 of The Finnish environment. Finnish Environment Institute, 2012. pp. 167–174.

[3] Autio, O., Toivonen, T., and Valpola, S. Etelä-Pohjanmaan suoselvityshanke. Loppuraportti. Etelä-Pohjanmaan liitto, Seinäjoki, 2013.

[4] Kondelin, H. Environmental impact of mire utilization. In: Lindholm, T. and Heikkilä, R., (eds.), Finland - land of mires, volume 23 of The Finnish environment. Finnish Environment Institute, 2006. pp. 193–204.

[5] Li, J. and Bioucas-Dias, J. M. Semisupervised Hyperspectral Image Segmentation Using Multinomial Logistic Regression With Active Learning. IEEE Transactions on Geosciences and Remote Sensing, 48(2010)11, pp. 4085–4098.

[6] Ruusuvuori, P., Manninen, T., and Huttunen, H. Image segmentation using sparse logistic regression with spatial prior. Proceedings of the 20th European Signal Processing Conference (EUSIPCO2012), Bucharest, Romania, August 27-30 2012. EURASIP. pp. 2253–2257.

[7] Boykov, Y., Veksler, O., and Zabih, R. Fast Approximate Energy Minimization via Graph Cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(2001)11, pp. 1222–1239.

[8] Cohen, F. S. and Wang, J.-Y. Part I: Modeling Image Curves Using Invariant 3-D Object Curve Models – A Path to 3-D Recognition and Shape Estimation from Image Contours. IEEE Transactions on Pattern Analysis and Machine Learning, 16(1994)1, pp. 1–12.

[9] Cohen, F. S., Huang, Z., and Yang, Z. Invariant Matching and Identification of Curves Using B-Splines Curve Presentation. IEEE Transactions on Image Processing, 4(1995)1, pp. 1–10.

[10] Wang, H. Fiber property characterization by image processing. Master's thesis, Texas Tech University, 2007. 82 p.

[11] Huttunen, H., Oinonen, H., Selinummi, J., Ruusuvuori, P., and Voipio, V. Polynomial pulp fiber modeling. Proceedings of the 20th European Signal Processing Conference (EUSIPCO2012), Bucharest, Romania, August 27-30 2012. EURASIP. pp. 2099–2103.

[12] Liimatainen, K., Heikkilä, R., Yli-Harja, O., Huttunen, H., and Ruusuvuori, P. Sparse logistic regression and polynomial modeling for detection of artificial drainage networks. Remote Sensing Letters, (2014). In revision.

[13] Moritz, H. Geodetic Reference System 1980. Journal of Geodesy, 74(1980)1, pp. 128–133.

[14] Zhen, S. H. Speed of the Continental Plates [WWW]. [Accessed on: 21.5.2014]. Available at: `http://hypertextbook.com/facts/ZhenHuang.shtml`.

[15] National land survey of Finland. Coordinate systems [WWW]. [Accessed on: 19.5.2014]. Available at: `http://www.maanmittauslaitos.fi/kartat/koordinaatit/`.

[16] Erol, B., Erol, S., and Celik, R. N. Precise geoid model determination using gps technique and geodetic applications. Proceedings of the 2nd International Conference on Recent Advances in Space Technologies. IEEE, 2005.

[17] National land survey of Finland. N2000 Valtakunnallinen korkeusjärjestelmä [WWW]. [Accessed on: 19.5.2014]. Available at: `http://www.maanmittauslaitos.fi/sites/default/files/N2000_Valtakunnallinen_korkeusjarjestelma.pdf`.

[18] Khorram, S., Koch, F. H., van der Wiele, C. F., and Nelson, S. A. C. Remote sensing. Springer, 2012. 134 p.

[19] Flood, M. Laser Altimetry: From Science to Commercial Lidar Mapping. Photogrammetric Engineering and Remote Sensing, 67(2001)11, pp. 1209–1218.

[20] Ring, J. The laser in astronomy. New Scientist, 18(1963)344, pp. 672–673.

[21] Zhou, W. An Object-Based Approach for Urban Land Cover Classification: Integrating LiDAR Height and Intensity Data. IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, 10(2013)4, pp. 928–931.

[22] National land survey of Finland. Laserkeilausaineisto (laser scanning materials) [WWW]. [Accessed on: 14.4.2014]. Available at: `http://www.maanmittauslaitos.fi/digituotteet/laserkeilausaineisto`.

[23] Isenburg, M. Lastools. Software (2013). [Accessed on: 1.10.2013]. Available at: `http://www.cs.unc.edu/~isenburg/lastools/`.

[24] Vilhomaa, J. and Laaksonen, H. Valtakunnallinen laserkeilaus – testityöstä tuotantoon. The Photogrammetric Journal of Finland, 22(2011)3, pp. 82–91.

[25] Li, Z., Zhu, Q., and Gold, C. Digital Terrain Modeling: Principles and Methology. CRC Press, 2005. 318 p.

[26] Gonzales, R. C. and Woods, R. E. Digital Image Processing. Pearson Education, Inc., 3rd edition, 2008. 954 p.

[27] Matlab® R2014a Documentation.

[28] Ojala, T., Pietikäinen, M., and Harwood, D. A comparative study of texture measures with classification based on feature distribution. Pattern Recognition, 29(1996)1, pp. 51–59.

[29] Ojala, T., Pietikäinen, M., and Mäenpää, T. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(2002)7, pp. 971–987.

[30] Jin, H., Liu, Q., Lu, H., and Tond, X. Face Detection Using Improved LBP Under Bayesian Framework. Proceedings of the Third International Conference on Image and Graphics, 2004.

[31] Hafiane, A., Seetharaman, G., and Zavidovique, B. Median Binary Pattern for Textures Classification. In: Kamel, M. and Campilho, A., (eds.), Image Analysis and Recognition, volume 4633 of Lecture Notes in Computer Science. Springer Berlin Heiderberg, 2007. pp. 387–398.

[32] Tan, X. and Triggs, B. Enhanced Local Feature Sets for Face Recognition Under Difficult Lighting Conditions. In: Zhou, S. K., Zhao, W., Tang, X., and Gong, S., (eds.), Analysis and Modeling of Faces and Gestures, volume 4778 of Lecture Notes in Computer Science. Springer Berlin Heiderberg, 2007. pp. 168–182.

[33] Nanni, L., Lumini, A., and Brahnam, S. Local binary patterns variants as texture descriptors for medical image analysis. Artificial Intelligence in Medicine, 49(2010), pp. 117–125.

[34] Heikkilä, M. and Pietikäinen, M. A Texture-Based Method for Modeling the Background and Detecting Moving Objects. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(2006)4, pp. 657–662.

[35] Mäenpää, T. The Local Binary Pattern Approach to Texture Analysis – Extensions and Applications. PhD thesis, University of Oulu, 2003. 78 p.

[36] Olivo-Marin, J.-C. Extraction of spots in biological images using multiscale products. Pattern Recognition, 35(2002)9, pp. 1989–1996.

[37] Duda, R. O., Hart, P. E., and Stork, D. G. Pattern Classification. Wiley-Interscience, 2nd edition, 2000. 680 p.

[38] Manninen, T. Predictive Modeling Using Sparse Logistic Regression with Applications. PhD thesis, Tampere University of Technology, 2014. 97 p.

[39] Friedman, J., Hastie, T., and Tibshirani, R. Regularization Paths for Generalized Models via Coordinate Descent. Journal of Statistical Software, 49(2010)6, pp. 1–22.

[40] Tibshirani, R. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society, Series B, 58(1996)1, pp. 267–288.

[41] Qian, J., Hastie, T., Friedman, J., Tibshirani, R., and Simon, N. Glmnet for matlab. Software (2013). [Included in pmtk3 package]. Available at: http://web.stanford.edu/~hastie/glmnet_matlab/.

[42] Guo, Z. and Hall, R. W. Parallel Thinning with Two-Subiteration Algorithm. Communications of the ACM, 32(1989)3, pp. 359–373.

[43] Niemistö, A., Dunmire, V., Yli-Harja, O., Zhang, W., and Shmulevich, I. Robust Quantification of In Vitro Angiogenesis Through Image Analysis. IEEE Transactions on Medical Imaging, 24(2005).

[44] Tuominen, J. and Lipping, T. Automatic Extraction of Road Elevation Map for Flood Rescue Operations. Proceedings of the 33rd International Symposium on Remote Sensing of Environment (ISRSE), Stresa, Italy, 2009.

[45] Astola, J. and Kuosmanen, P. Fundamentals of Nonlinear Digital Filtering. CRC Press, New York, 1997. 276 p., pp. 105-107.

[46] Xudong, L., Xuedong, Z., and Youchuan, W. A Kind of Filtering Algorithms For LiDAR Intensity Image Based on Flatness Terrain. Proceedings of the International Symposium on Spatio-temporal Modeling, Spatial Reasoning, Analysis, Data Mining and Data Fusion, Beijing, China, 2005.

[47] Probabilistic modeling toolkit version 3 (pmtk3). Software (April 2011). [Accessed on: 1.12.2013]. Available at: http://pmtk3.googlecode.com/files/pmtk3-2april2011.zip.

[48] Fawcett, T. An introduction to ROC analysis. Pattern Recognition Letters, 27(2006), pp. 861 – 874.

APPENDIX 1: FEATURES FOR DITCH DETECTION

Table 1: Features for ditch detection. There were 1065 features in total.

Feature	Parameters	Values
Filtering		
Edge emphasizing filtering Gaussian low-pass filtering Gaussian filtering difference	kernel size	3:5:53
Average filtering difference Circular average filtering difference	kernel size 1, kernel size 2	3:2:33, ks1 + 2:2:12
Thresholding and h-maxima transformation		
Masking with h-maxima transformation	size of h	3:5:53
Morphological transformations		
Top-hat transformation Bottom-hat transformation	kernel size	3:5:53
Local statistical properties		
Standard deviation	kernel size	3:4:51
Variance Third image moment Fourth image moment Spectral energy density Entropy	kernel size	7:4:23
Range	kernel size	3:4:51
Local binary patterns		
Rotation invariant LBP Rotation invariant ILBP Rotation invariant MBP	radius, number of points	1:1:20, 8:4:16
LTP ILTP RBP	radius, number of points, threshold	1:1:10, 8:4:16, 0.05:0.05:4
Local contrast	radius number of points	1:0.5:5 8:4:16
Multi-resolution LBP	scale	2:1:6
Sum of multi-resolution LBPs	scale	4:1:8
Gaussian filter bank difference	scale	3:1:6
Division of LBP and local contrast	radius number of points	1:0.5:5 8:4:16
Wavelet decomposition		
Wavelet decomposition images	depth plane	1:1:3
Gradient magnitude		
Gradient magnitude image		

APPENDIX 2: FEATURES FOR ROAD DETECTION

Table 2: Features for road detection. There were 1468 features in total.

Abbreviations of input images:
DTM Digital terrain model
DSM Digital surface model
LI LiDAR intensity image
GO Grayscale orthophotograph

Feature	Parameters	Values	Images
Filtering			
Edge emphasizing filtering	kernel size	3:5:53	DSM, LI, GO
Gaussian low-pass filtering	kernel size	3:5:53	DSM, LI
Gaussian filtering difference	kernel size	3:5:53	DSM, LI
Thresholding and h-maxima transformation			
Thresholding	threshold	20:1:70	LI
Thresholding	threshold	100:5:200	GO
Masking with h-maxima transformation	size of h	3:5:53	DSM, LI
Morphological operations			
Top-hat transformation	kernel size	3:5:53	DSM, LI, GO
Bottom-hat transformation	kernel size	3:5:53	DSM, LI, GO
Local statistical properties			
Mean	kernel size	3:4:23	DSM, LI, GO
Standard deviation	kernel size	3:4:51	DSM, LI, GO
Variance	kernel size	3:4:23	DSM, LI, GO
Third image moment	kernel size	3:4:23	DSM, LI, GO
Fourth image moment	kernel size	3:4:23	DSM, LI, GO
Spectral energy density	kernel size	3:4:23	DSM, LI, GO
Entropy	kernel size	3:4:23	DSM, LI, GO
Range	kernel size	3:4:51	DSM, LI, GO
Local binary patterns			
LBP	radius number of points	1 8	DSM, LI
ILBP (rotation invariant)	radius number of points	1:1:20 8:4:16	LI, GO
MBP (rotation invariant)	radius number of points	1:1:20 8:4:16	LI, GO
LTP	radius number of points threshold	1:1:20 8,16 0.05:0.1:0.5	DSM, LI, GO
ILTP	radius number of points threshold	1:1:20 8:4:16 0:0.05:0.4	LI
Local contrast	radius number of points	1:0.5:2 8:4:16	DSM, LI
Multi-resolution LBP	scale	2:1:6	DTM, DSM, LI, GO
Sum of multi-resolution LBPs	scale	4:1:8	LI
Gaussian filter bank difference	scale	4:1:6	LI, GO
Division of LBP and local contrast	radius number of points	1:0.5:2 8:4:16	LI
Sum of multi-resolution ILBPs	scale	0:1:2	LI
Wavelet decomposition			
Wavelet decomposition images	plane	1:1:3	DSM, LI, GO
Gradient magnitude			
Gradient magnitude image			LI