Spatial Features for Tree-crown Classification from 3-cm Resolution CIR Imagery

Linus Närvä

August 26, 2015

Key words: tree-crown classification, single tree remote sensing (STRS), feature selection, support vector machine (SVM), Gray-level co-occurrence matrix (GLCM), Haar wavelets.

Abstract

Species composition is one of the most important factors in forest inventory. Species is an extrinsic property of an individual tree. Therefore, a feasible remote-sensing approach to estimate the species composition of a forest population is to classify individual tree-crowns. On the single-tree level, the problem of tree-crown classification from aerial imagery has been studied since the beginning of the nineties. However, species classification remains a bottleneck in remote-sensing-based forest inventory.

Many of the studies in tree-crown classification deals with *multispectral features*, i.e. measures of intensity in different spectral bands (e.g. colours). Such features suffers drawbacks of being sensitive to factors such as viewing geometry, sun angle, time of day, and time of year. Another possible approach is to base tree-crown classification on *spatial features* (Brandtberg, 2002; Erikson, 2004), that measures local changes in pixel intensity.

This primary objective of this study is to evaluate a large set of spatial features and to compare these with some multispectral features, for the problem of discriminating the tree species spruce, pine, and birch. In total 366 features were evaluated on a dataset of 875 trees from a boreal forest in southern Sweden.

The spatial features are derived by techniques of image morphology, image gradients, and from textural measures based on second-order statistics. Specifically, the textural measures are based on the Gray-level co-occurrence matrix (GLCM) and on Haar wavelets. For the Support Vector Machine (SVM) classifier the relative importance of each feature was established by an SVM-based feature ranking algorithm proposed in Guyon et al. (2002). In order to investigate what spectral information is necessary, the classification for the best features was measured in the respective subsets; all features, spectral features, spatial features, and spatial features in a single colour channel.

The results demonstrates that the most powerful features were either based on the GLCM, or on Haar wavelets, possibly because those features analyses the tree-crown imagery at multiple scales. Another result was that excluding spectral features did not cause a substantial drop in classification accuracy if sufficiently many best-features were used. For smaller best-feature sets, the spectral features can be replaced by spatial features if the total number of best-features is increased. However, the spectral information from utilizing multiple spectral bands was found to be necessary even for spatial features. That is, the classification accuracy from limiting spatial features to a single colour channel was impaired substantially, compared to when all three CIR channels was utilized. The compound classification accuracy is measured to 76%, a figure that is comparable to other studies using spatial features for tree-crown classification.

The study shows that spatial features are powerful for imagery-based tree-crown classification at 3 cm resolution.

Contents

1 Introduction

1.1 Background

Forest inventory is the process of estimating properties of total forest populations (West and West, 2009). In particular, timber volume and species composition are two of the most important forest parameters (Walter, 1998). What method of forest inventory that is suitable is determined by factors such as: the delineation of the population, the definition of individuals in the population, what properties are to be measured, and what method of analysis that is used[1] (West and West, 2009). Manual measurements are however cumbersome. Even for small forests it becomes intractable to extract whatever information the owner wants to know about the population, by manually measuring every tree (West and West, 2009, p. 91). Manual small scale forest inventory is therefore primarily concerned with measuring tree properties of *samples* of the population. The samples are often a set of trees within predetermined area plots called *stands*. The measurements of the stands can then be generalized to the entire population, using statistics and sampling theory (West and West, 2009, p. 91).

Large-scale forest inventory requires remote sensing techniques, in order to cover large areas at a relatively low cost (compared to manual ground measurements) (Shao and Reynolds, 2006, p. 4). Forest remote sensing is an umbrella term for analysis- and data acquisition- techniques from aerial surveys (Walter, 1998). Two major data acquisition methods of today are aerial imagery and airborne laser scanning (ALS) (Nordkvist and Olsson, 2013). The *Area based method* (introduced in Scandinavia 2002) is a major methodology for conducting forest inventory based on remote sensing. In the area based method, the whole population is surveyed (usually by ALS). The measurements of the survey is calibrated to manual ground measurements. Parameters of interest are then estimated using different methods of analysis. The Area based method mainly estimates parameters related to tree size. However, a particularly important property that is not estimated by the Area based method is species composition (Nordkvist and Olsson, 2013). In fact, to this day, no published results has produced a classification accuracy that is suitable for operational use (Nordkvist and Olsson, 2013). Species classification remains a bottleneck in forest inventory based on remote sensing (Korpela et al., 2010).

[1] In this article it is natural to think of the trees as individuals, and the evaluated stands as constituting the population.

1.2 Related work

Operational methods for forest inventory, based on remote sensing (e.g. the area based method), mainly estimates forest parameters on the *stand level* (for groups of trees). However, certain important forest parameters are extrinsic to individual trees and therefore cannot be directly measured on the stand level. One particular such parameter is species. Given remote sensing data of sufficient detail, the analysis can be conducted on a single-tree level (Nordkvist and Olsson, 2013). This sub-field of forest remote sensing that models individual trees is called *Single tree remote sensing* (STRS).

An important sub-problem that arises in STRS is *individual tree crown delineation*[2]. ALS data has proven valuable for automatically delineating individual tree crowns (see e.g. Persson et al. 2002). For a comparison of different ALS based tree crown delineation techniques, see Kaartinen et al. (2012); Vauhkonen et al. (2011). It is also possible to use aerial 2D imagery for tree crown delineation. However, image based delineation algorithms in general suffers from commission errors. The reason of the commission errors is that in open areas, structures in the lower forest floor, such as shrubs or clumped vegetation, are mistaken for trees. If however height information is available, these commission errors can be removed (Leckie et al., 2003a). A review of 2D imagery based automatic tree crown delineation is found in Brandtberg and Warner (2006). Aerial imagery is not necessarily limited to 2D information though. It is possible to construct a *digital canopy model* (DCM) of canopy elevation, using photogrammetry on overlapping aerial images (Nordkvist and Olsson, 2013). To obtain a DCM from ALS data is straightforward, because laser scans produces 3D information directly. DCMs has shown especially useful for automatic tree crown delineation (Nordkvist and Olsson, 2013).

The STRS community today regards aerial imagery mainly as a complement to ALS. In particular, this is the case for species classification (Leckie et al., 2003a; Packalén et al., 2009). The advantage of ALS stems from its ability to capture structural information, because it produces highly detailed 3D point clouds of the scanned environment. A specially important property of ALS is that individual rays passes through the forest canopy, capturing the ground and the internal structure of the trees (in contrast to photogrammetry that only captures the uppermost surface of the forest canopy) (Nordkvist and Olsson, 2013). The benefit of 2D imagery, on the other hand, is its ability to capture multispectral information and spatial details useful for species and health assessments (Leckie et al., 2003a; Brandtberg and Warner, 2006).

Even though research efforts in species classification based on ALS alone, demonstrates the usefulness of ALS data in species classification[3] (e.g. Brandtberg et al., 2003; Brandtberg, 2007; Ørka et al., 2007, 2009; Li et al., 2010; Korpela et al., 2010), the benefit of including aerial imagery must not be disregarded. That is, because aerial imagery

[2] A note on terminology. Tree-crown delineation is the accepted term in STRS literature, for determining which pixels/points belong to respective tree-crowns. However, in image processing, image analysis, and computer vision in general, the corresponding term is tree-crown segmentation (Gonzalez, 2008, chapter 10).

[3] It is important to note that the features used in ALS based species classification are note exclusively structural (i.e. derived from the geometry of the 3D point cloud), but also based on properties of the laser echoes, e.g. the mean and standard deviation of echo intensities (Ørka et al., 2007).

in addition to ALS contributes to classification accuracy by a significant amount (Packalén et al., 2009). Examples of laser-imagery hybrid systems for individual tree species classification are found in Holmgren et al. (2008) and Ke et al. (2010), both which concludes that the synthesized data yields better classification accuracy than to use only laser features, or only image features. In their analysis, Holmgren et al. (2008) includes the classification accuracies from the respective feature sets used separately. Interestingly, the resulting classification accuracies are comparable. However, the performance of the image feature, specifically the mean pixel intensity per colour band of the tree crown, was found to vary by a significant amount between different seasons.

On the single-tree level, species classification based on aerial imagery have a longer history than methods based on ALS. Early work in species classification from aerial imagery is found in Gougeon et al. (1989); Pinz and Bischof (1990); Pinz et al. (1993). In general, aerial imagery data can be separated into three categories of data; multispectral, multitemporal, and spatial (Key et al., 2001). Multispectral information correspond to the number of spectral bands used (e.g. colours). Multitemporal information correspond to the number of dates (time points) the the scene has been photographed. Spatial information correspond to local changes in pixel intensity. Generally, the spatial information increases with increasing resolution.

Important work in individual tree species classification from multispectral features is found in Gougeon (1995). Gougeon (1995) evaluated seven spectral features for classifying five species of conifers[4] in Canada. The study found two interesting results. Firstly, the simplest features; the mean intensity values of the respective spectral bands contributed the most to the classification accuracy. Secondly, repeating the experiment (ten times), using different training data, resulted in a large variation of classification accuracy of up to 22 pp[5].

The idea of utilizing multitemporal data in individual tree species classification is that species-relevant phenological changes can be measured by collecting the imagery at optimal dates. Moreover, the phenological changes are often changes in spectral properties (e.g. leaves changing colour in fall). This observation motivates to utilize the combination of multitemporal and multispectral data (Key et al., 2001). Key et al. (2001) therefore investigated the relative contribution to species classification of adding spectral bands versus adding dates. The experiment uses data from a deciduous forest[6] in West Virginia. The study discovered that the contribution to classification accuracy per spectral band is slightly greater than the contribution per date.

The last category of image data is spatial data. Examples of spatial data include structure, texture[7], and morphology. Human visual interpretation is to a large extent based on the interpretation of spatial information (Brandtberg and Warner, 2006). It is therefore sensible to assume that there should exist spatial descriptors that are able to discriminate between tree species. Indeed, comprehensive catalogues of spatial descriptors such as

[4]Barrträd in Swedish.

[5]Percentage points (pp) is the unit of arithmetic difference between percentages. For example, the difference between 1% and 5% is 4 pp.

[6]Lövskog in Swedish.

[7]Strictly speaking, only second-order texture is considered to be spatial information. See section 2.5.

Fournier et al. (1995) have been produced for the purpose of species classification, from manual interpretation of aerial images. However, to utilize spatial data in automatic classification has proven difficult. Specifically, it is hard to specify exactly what local changes to look for, and at what scale to look for them (Brandtberg and Warner, 2006). As a consequence, much of the work in species classification from spatial features also uses features derived from multi-spectral information (e.g. Meyera et al., 1996; Brandtberg, 2002), or indirectly uses spectral information by deriving spatial features from multiple spectral bands (e.g. Erikson, 2004). In some cases, the simple multispectral features contribute more to classification than the spatial features (e.g. Brandtberg, 2002). Still, there are good reasons to omit spectral features in species classification. The problem of basing features on absolute intensity properties (e.g. mean pixel intensity) is that such features are sensitive to changes in illumination. For species classification from aerial images, error sources include: different atmospheric conditions, or weather; different times of day/year; and differences in viewing geometry for trees at different positions within the image (Puttonen et al., 2009; Korpela et al., 2011; Lillesand et al., 2004). Furthermore, the spectral signatures of trees varies within species (Leckie et al., 2005). Also, the spectral signature of trees (even coniferous species) varies with time of year (Guyot et al., 1989). Guyot et al. (1989) identifies a comprehensive list of factors affecting the spectral signatures of forest canopies.

To the best of the authors knowledge, textural measures based on second-order statistics has never been tried in species classification on the single-tree level. However, on a stand level, Franklin et al. (2000, 2001) investigated if features derived from the *Gray-level co-occurrence matrix* (GLCM) in addition to multispectral features can be used to improve classification accuracy, over multispectral features alone. The study found that the incorporation of texture yielded a significantly greater improvement at a finer scale (sub-meter resolution) compared with the improvement at a coarser scale. Textural measures based on first-order statistics are utilized at the single-tree level in (Meyera et al., 1996), and in (Leckie et al., 2003b). Specifically, in the form of intensity standard deviation of all pixels within the tree crown. The results of the two studies where ambiguous. Leckie et al. found that texture did not contribute to classification beyond using just spectral information. In contrast, Meyera et al. found an improvement when including texture in *principal-component-analysis-derived* features. Another first order textural measure; the entropy of pixel intensities within the tree crown, was utilized at the single-tree level in (Brandtberg, 2002). The contribution to classification accuracy was found to be modest.

Morphological features in individual tree species classification was introduced by Brandtberg (1997, 2002). The introduced feature aims to capture the radial branch pattern of a Norwegian spruce crown. In later work by Erikson (2004), four morphological features were suggested. Each of these features is supposed to single out a specific tree class (Scots pine, Norwegian spruce, birch, and aspen). The measures captures properties apparent from inspection of tree images of respective species.

A final class of spatial features for individual tree species classification describes the geometry of the tree crowns. For example, in (Brandtberg, 2002), two boundary descriptors and one regional descriptor was defined. The former two features were derived from the curvature of the boundary pixels between the sunlit and shadowed regions of the tree

crown. The last feature is the relative crown area (see also Brandtberg, 1998).

The studies mentioned this far evaluates classification from a relatively small subset of well-chosen features, where each feature is expected to contribute information. An alternative approach is to use a larger set of features, where only a subset of features is expected to be useful, and to search for the useful features by means of feature selection, or data mining. Boman (2013) utilizes an SVM-based method of feature selection, developed in Guyon et al. (2002), to evaluate (mainly) textural features for tree classification. However, the classification problem treated in Boman (2013) is fundamentally different from the one in this study. Boman (2013) uses terrestrial imagery and classifies 128×128 sub-images into the classes *spruce*, *pine*, and *ground*.

1.3 Objective

Brandtberg (2002) concludes that it is a difficult, unsolved problem in tree-crown classification to find appropriate descriptors that contribute significant information to the classifier. Thirteen years has passed since then, but to the best of the author's knowledge, Erikson (2004) is the only study, after Brandtberg (2002), that uses spatial features in their own right.

Therefore, the primary objective is to systemically evaluate a large set of features for discriminating three classes of trees in a Swedish, boreal forest. The tree classes are specifically; *spruce*, *pine*, and *deciduous trees*[8]. Primarily, spatial features that represent wide categories of spatial information are considered, but a smaller set of spectral features are also included for comparison. Because spatial features in general may have advantages over spectral features, the relative contribution to classification performance of respective feature subset (spatial and spectral) is of particular interest. What kind of spectral information is required for classification and what kind of spectral information the classifier can do without, is subject to analysis.

Spatial features is defined as local changes in pixel intensity. Observe that this definition excludes textural measures based on first-order statistics, such as the standard deviation in pixel intensity within the tree crown. Such measures are global in the sense that they do not take into account where particular intensities occur. The distinction between the two texture classes is further clarified in section 2.5. To avoid ambiguity, *spectral features* are here defined as measures of intensity that are global to the tree-crown. This definition includes textural measures based on first-order statistics.

[8] In Swedish: gran, tall, and lövträd, respectively

2 Theory

2.1 Symbols and notation

This sections introduces symbols and notation that is used to describe theory in the remainder of the this text. Scalars and functions are denoted by lower case letters, e.g. a, b, c. For functions in general, the symbols f, g, h are preferred. Vectors are denoted by small letters in bold font, e.g. $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$. Matrices and digital images (expressed as matrices) are denoted by capital letters, e.g. X, Y. Sets and random variables will also be denoted by capital letters. For random variables, Z is preferred. Vectors and matrices are referenced in the ordinary manner. For example the i'th element of the vector $\boldsymbol{v}$ can be written $\boldsymbol{v}(i)$, or just v_i. Similarly, for a matrix X the element on the i'th row and j'th column is written $X(i, j)$, or just x_{ij}. The symbols i, j are in general reserved for indices. For matrices, i refers to the row index and j to the column index. Unless otherwise stated, indices are assumed to be positive integers, within bounds of the vector or matrix in context. For example, if the vector $\boldsymbol{v}$ has n elements, then for v_i it can be assumed that i is an integer such that $1 \leq i \leq n$.

Exceptions from these rules are made when there are well established symbols that denotes a particular entity. For example, *entropy* will be denoted by S, even though it is a scalar entity.

2.2 Digital image fundamentals

This section introduces a theoretical framework of *digital images*, suitable for this article. Because continuous images and physical photographs are not considered in this work, a digital image may be referred to as just an *image*. The section is based on Gonzalez (2008, chap. 2.4.2, and p. 120).

A $m \times n$-pixel *digital grey-image* can be represented by a $m \times n$ matrix X of discrete *pixel-intensity values*, or simply *intensities*. Furthermore, intensities must be restricted to a closed interval of non-negative numbers I, starting at 0. Then, if l denotes the number of intensity-levels, the interval of possible intensities can be expressed as $I = \{\forall r \in \mathbb{Z} | 0 \leq r < l\}$. Similarly, a *digital colour image* with $m \times n$ pixels and c colour-channels, can be represented by an $m \times n \times c$ three-dimensional matrix[9].

In some cases, it is more convenient to represent the image as a function $f : S \to I$, where the *spatial domain* correspond to set of discrete coordinates inside the image, i.e. $S = \{1, 2, ...m\} \times \{1, 2, ...n\}$. This representation is equivalent to $f(i, j) = X(i, j)$ for grey-images, and similarly $f(i, j, k) = X(i, j, k)$ for colour images.

[9]The colour images defined here are more general than in everyday language. Any number of colour-channels is allowed and colour-channels are not bound to specific spectral bands such as red, green, and blue.

2.2.1 Image histograms

The *image histogram* h of the grey-image X is a discrete function

$$h : I \to [0, 1] \tag{2.2.1}$$

that maps a pixel intensity to the *fraction* of pixels in X of that intensity. Formally, this is written

$$h(r) = \frac{n_r}{mn}, \tag{2.2.2}$$

where $r \in I$ is a particular intensity level and n_r is the count of occurrences of r in X. An important implication of (2.2.2) is that the sum of h's over its domain I is exactly equal to 1, i.e.

$$1 = \sum_{r \in I} h(r). \tag{2.2.3}$$

Because h satisfies (2.2.1) and (2.2.3), it can be interpreted as a *probability mass function*. That is, corresponding to h there will be a *discrete random variable* Z of pixel-intensity-observations such that h is the probability mass function of Z, i.e. $P_Z(r) = h(r)$. An observation of Z is interpreted as the intensity of an uniformly-randomly chosen pixel in X. Also, note that the sample space of Z is equal to I, i.e. $\Omega_Z = I$. These results are important because they mean that statistical methods can be utilized to analyse the intensity distribution within an image.

It is sometimes necessary to reduce the number of intensity levels in the image while conducting analysis. An 8-bit image will have 256 grey-levels, which in some applications may require too much computational performance. A remedy is to reduce the number of intensity levels setting the intensity to $\hat{r} = \lfloor sr \rfloor$, for the scale factor $0 < s$. For example $s = 0.5$ will effectively halve the number of levels. This operation is called *quantization*[10] (Gonzalez, 2008, chap. 2.4). Figure 2.2.1 shows some quantized image histograms.

[10] Quantization also refers to the process of sampling and digitizing a continuous image, to obtain a digital image (Gonzalez, 2008, chap. 2.4).

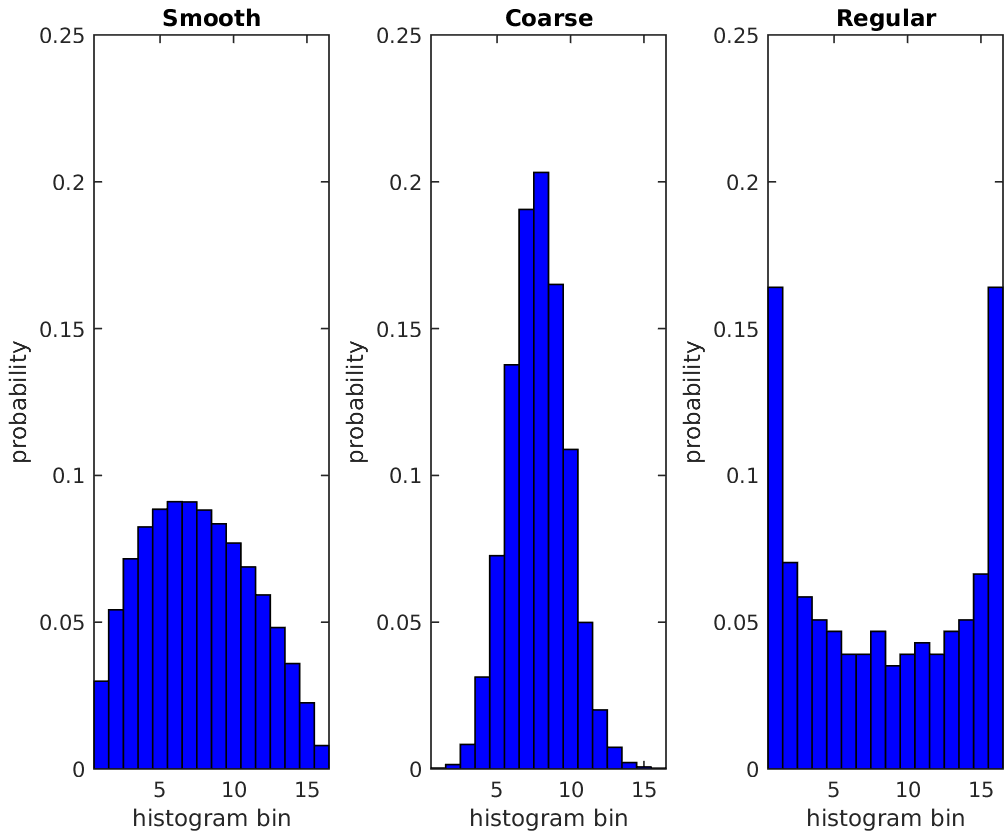


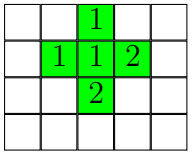
Figure 2.2.1: Sixteen-bin histograms of the images in figure 2.5.1 (the scale factor is $s = 1/16$). The histogram of the smooth image is skewed to the left because it has more pixels of lower intensity. The histogram of the coarse image roughly correspond to a symmetric bell-curve. The first two histograms are centred near the middle indicating dominance of grey pixels. However, the last histogram, which correspond to the regular image, is U-shaped. This indicates dominance of black and white pixels, respectively.

2.3 Image filtering

An image filter is a transformation T that is defined on a pixel-neighbourhood, i.e.

$$Y = T(X), \tag{2.3.1}$$

for the $m \times n$ original grey-image X and the $m' \times n'$ *filtered image* Y. For example, T may be defined to apply the function g on a pixel and the pixels 4-connected neighbours. Applying $Y = T(X)$ means that for all pixels in X, the corresponding pixel in Y has intensity $r_Y = g(r, r_1, r_2, r_3, r_4)$, where r is the intensity of the particular pixel in X and r_1,r_2,r_3,r_4 the intensities of its neighbours. This is illustrated in figure 2.3.1. Depending on the definition of the neighbourhood, the size of Y may be smaller than the size of X. Depending on g, the interval of intensities of Y may differ from X, i.e. it is possible that $I_X \neq I_Y$. In fact the filtered result is not even guaranteed to be a digital image, according to the definition above.



(a) X

(b) Y

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

(c) T

Figure 2.3.1: Illustration of image filtering $Y = T(X)$, where T is defined for a 4-connected neighbour and applies $h(r, r_1, r_2, r_3, r_4) = r + r_1 + r_2 + r_3 + r_4$, to each pixel in X to yield Y. Pixels that where involved for computing the value of one pixel in Y are highlighted. Note that Y is smaller than X, because the neighbourhood is undefined at the borders of X. Also note that Y is not restricted to the same intensity interval as X, i.e. $I_X \neq I_Y$ is possible. In practice, both these issues must be dealt with. Furthermore, since T is a linear filter, it can be expressed as a weight matrix (c).

The most common type of filters are *linear filters*. A linear filter can be expressed as a weight matrix, e.g.

$$F = \frac{1}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Linear filtering involves dragging the filter over the image, and compute the weighted sum of pixels at each position in the spatial domain of X where the filter *fits*[11]. In this article, the notation

$$Y = X * F, \tag{2.3.2}$$

is used to denote the linear filtering of the image X by the filter F. As a basic example

[11] In some literature, this version of filtering is commonly referred to as *correlation* and the filter matrix is referred to as a *kernel* (Gonzalez, 2008, chap. 3.4.2).

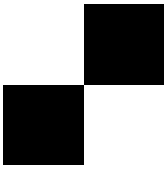
consider the image

$$X = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

and the filter F defined above. The result from filtering X with F is

$$X * F = \begin{bmatrix} 1 & .5 & 0 \\ .5 & .5 & .5 \\ 0 & .5 & 1 \end{bmatrix}.$$

A graphical illustration of this operation is shown in figure 2.3.2. For more information on image filtering, see Gonzalez (2008, chapter 3).



(a) Original

$$\frac{1}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

(b) Filter

(c) Result

Figure 2.3.2: Illustration of the the filtering with the sample F and X from the introductory text of section 2.3. The filter in (b) can be understood as a smoothing filter because it has removed the rapid intensity changes between adjacent black and white pixels. In the result (c) no white pixel has any black neighbours, and vice versa for black pixels.

2.3.1 Image derivatives

Image derivatives can be used to extract local intensity changes, or details of an image. For functions in general, change is represented by the derivative. For a one dimensional discrete function $f : \mathbb{Z} \to \mathbb{Z}$, the first-order derivative can be defined as

$$\frac{df}{dx} = f(x+1) - f(x). \tag{2.3.3}$$

Similarly, the second-order derivative can be defined as

$$\frac{d^2 f}{dx^2} = f(x+1) + f(x-1) - 2f(x). \tag{2.3.4}$$

Because images can be represented as functions of two variables $f(x, y) : \mathbb{N} \to I$, the concepts of derivatives apply. Because the image is two-dimensional the derivatives can be directional, e.g. measure the change in the x-axis, or y-axis, respectively. Alternatively, *isotropic* derivatives measures change in all directions. The simplest isotropic derivative is the Laplacian (Gonzalez, 2008, p. 160). For two variables, it is defined by

$$\bigtriangledown^2 f = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}. \tag{2.3.5}$$

Given the previous definition of the discrete second order derivative, the discrete Laplacian can be expressed as

$$\bigtriangledown^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y),$$

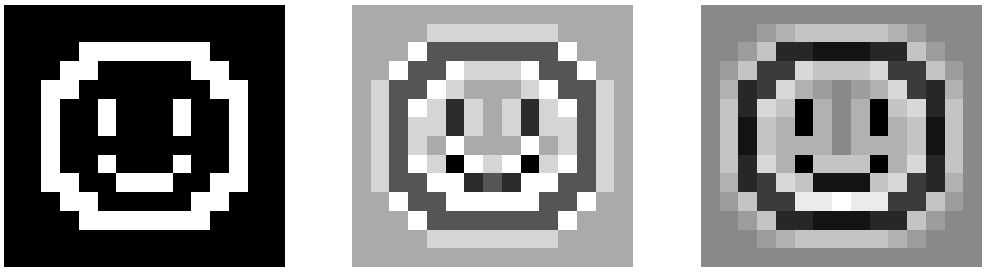
which correspond to the linear filter

$$L_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{2.3.6}$$

Because this filter measures change relative to the 4-connected neighbours, it is referred to as the 4-connected Laplacian. Alternative the 8-connected Laplacian, defined as

$$L_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \tag{2.3.7}$$

can be used to also take diagonal-wise change into account. The difference between the two is illustrated for a small image in figure 2.3.3. A more natural image and its L_8 filtered version is seen in figure 2.3.4.



(a) Original *(b) L_4-filtered* *(c) L_8-filtered*

Figure 2.3.3: Example of the 4-connected and 8-connected Laplacian filter, applied to a small image.

Anisotropic filters can be based on first order derivatives. Typically the filters used correspond to the principal axes, i.e. the filter D_x correspond to the x-wise derivative and the filter D_y correspond to the y-wise derivative. The filtered results

$$G_x = X * D_x, G_y = X * D_y, \tag{2.3.8}$$

are referred to as *image gradients*. As an example of simple directional derivatives, consider the filters

$$D_x = \begin{bmatrix} -1 & 1 \end{bmatrix}, D_y = D_x^T = \begin{bmatrix} -1 \\ 1 \end{bmatrix}. \tag{2.3.9}$$

The gradients computed from these respective filters are presented in figure 2.3.5. In general, the y-wise filter is the x-wise filter rotated by $90°$, or $D_y = D_x^T$. Therefore, it is

sufficient to specify D_x. Other examples of image gradients are (in all cases the x-wise version is given): the Roberts gradient

$$D_{roberts} = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \tag{2.3.10}$$

the Prewitt gradient

$$D_{prewitt} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \tag{2.3.11}$$

and the Sobel gradient

$$D_{sobel} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}. \tag{2.3.12}$$

An important detail is that the shape of the filter effectively determines the positions relative to the pixels, where the gradients are computed. $D_{roberts}$ evaluates the gradient at the corners of pixels, because the centre of the filter is between the four cells. $D_{prewitt}$, D_{sobel} on the other hand, evaluates the gradient at pixel centres of all pixels that are not on the image border.

Given the gradients G_x, and G_y, two useful measures at each pixel is the *magnitude of change* and the *direction of change*. Magnitude of change is defined as

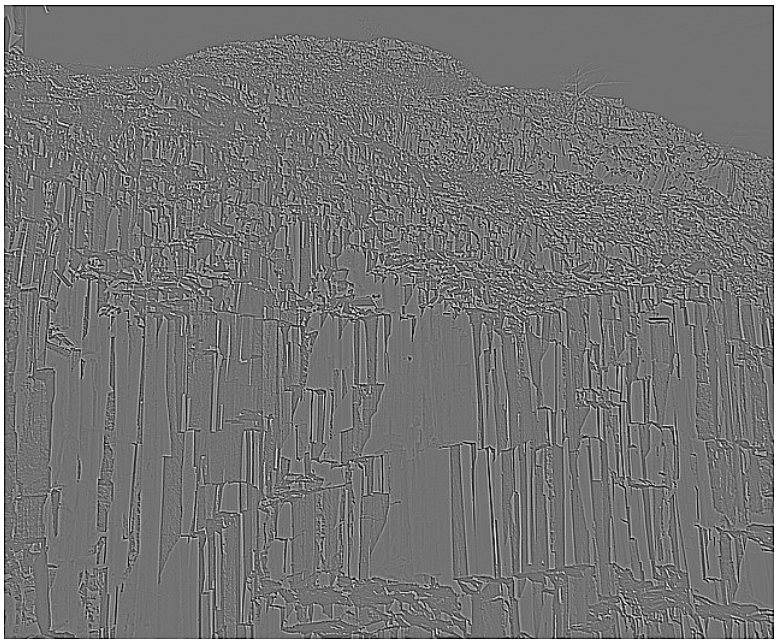
$$m_{ij} = \| \begin{bmatrix} G_x(i,j) \\ G_y(i,j) \end{bmatrix} \|_2 = \sqrt{(G_x(i,j))^2 + (G_y(i,j))^2}, \tag{2.3.13}$$

and direction of change (expressed as a unit vector) is defined as

$$d_{ij} = \frac{1}{m_{ij}} \begin{bmatrix} G_x(i,j) \\ G_y(i,j) \end{bmatrix}. \tag{2.3.14}$$

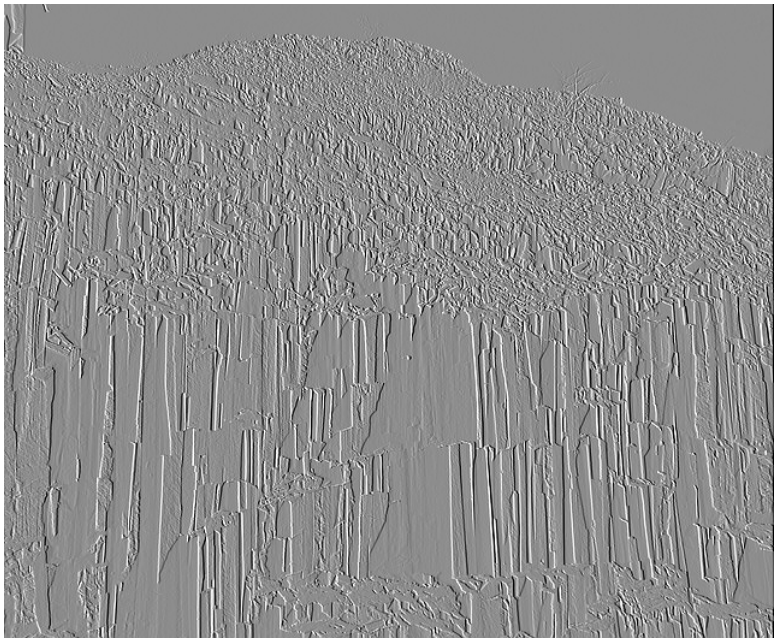
For more information on image derivatives, see Gonzalez (2008, chapter 3.6).

(a) Original

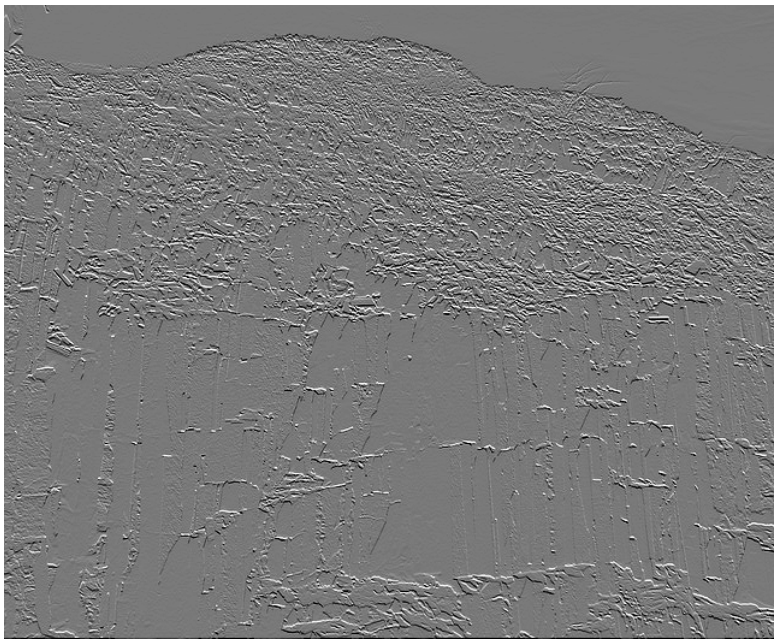


(b) L_8-filtered

Figure 2.3.4: L_8 is used to filter a natural image. On an intuitive level, the filter has extracted the edges, or contours of regions within the original image. Lightness and darkness information of regions have been discarded. The original image was provided by Pixbay, `http://pixabay.com`.



(a) G_x



(b) G_y

Figure 2.3.5: Illustration of image gradients, using the elementary filters D_x*,* D_y *from equation* (2.3.9)*, and the original image from figure 2.3.3a, to generate the directional derivatives, or image gradient* G_x*,* G_y*. The diagonal cracks are clearly distinguishable in* G_x*, but not seen in* G_y*.*

2.4 CIR imagery

CIR is shorthand for Colour-infra red. Specifically, CIR imagery combines colour bands with the the portion 0.7 μm to 0.9 μm of the near-infra red band (NIR). This is achieved by substituting the RGB channels in the image by the CIR spectral bands. NIR is substituted for red, red is substituted for green, and green is substituted for blue (the blue spectral band is dropped). Vegetation generally reflects much more in the NIR band than it does in the colour bands. In CIR imagery, vegetation therefore appears in varying tones of red (Lillesand et al., 2004, p. 91-96).

2.5 Introduction to texture

An important class of spatial features of an image region is based on its *textural* content. While no formal definition can be found in literature, texture intuitively describes properties such as *smoothness*, *coarseness*, and *regularity* of an image region (see figure 2.5.1). In image processing, three canonical approaches to texture are *statistical*, *structural*, and *spectral*. Statistical approaches are based on measuring statistical properties of pixel intensities, and mainly tries to capture properties such as smoothness, coarseness, granularity etc. Structural approaches focuses on the arrangement of image primitives. Finally, spectral approaches operates in the frequency domain (Fourier spectrum) and attempts to capture regularity (Gonzalez, 2008, chap. 11.3.3).

(a) SMOOTH *(b)* COARSE *(c)* REGULAR

Figure 2.5.1: Illustrations of the three texture classes.

Statistical approaches to texture can utilize first-, or second-order statistics (Aggarwal and Agrawal, 2012). In this article, these two approaches are referred to as *first-order texture* and *second-order texture*[12]. First-order textural measures are derived from the image histogram and encompasses measures such as standard deviation s, central moments μ_n's, and entropy S.

The *central moment* of order n (where $n \in \mathbb{N} \cup \{0\}$) is defined as (Gonzalez, 2008, p. 828)

$$\mu_n(Z) = \sum_{z \in \Omega} (z - m(Z))^n p(z), \tag{2.5.1}$$

[12]The terms are however not established in literature.

where Z is a random variable of pixel-intensity observations (as defined in section 2.2.1), whose sample space Ω is the set of intensity levels; and m is the mean, given by (Gonzalez, 2008, p. 828)

$$m(Z) = \sum_{z \in \Omega} zp(z). \tag{2.5.2}$$

Note that from these definitions it follows that $\mu_0 = 1$ and $\mu_1 = 0$. The most common central moments are: variance $\sigma^2 = \mu_2$[13], skewness μ_3, and kurtosis μ_4 (Aggarwal and Agrawal, 2012). A fundamental problem with these moments is that they are scale dependant. For example, skewness will take different value depending on if the same data is presented in meters or centimetres. The effect of scale can be eliminated completely by *standardizing* the moments (Ramsey et al., 2002). The *standardized moments* are defines as:

$$\hat{\alpha}_n(Z) = \frac{\mu_n}{\mu_2^{\frac{n}{2}}}, \tag{2.5.3}$$

or equivalently

$$\hat{\alpha}_n(Z) = \frac{\mu_n}{\sigma^n}, \tag{2.5.4}$$

where σ is the n-normalized standard deviation[13]. Note that these definitions yield $\hat{\alpha}_0 = 1$ and $\hat{\alpha}_1 = 0$ like before, but also $\hat{\alpha}_2 = 1$. In the remainder of this article, unless otherwise stated, the term *central moments* will refer to the standardized central moments. Additionally, skewness and kurtosis will refer to $\hat{\alpha}_3$ and $\hat{\alpha}_4$, respectively.

Another useful first-order textural measure is the *entropy* of the image histogram (utilized e.g. in Brandtberg, 2002). The definition of entropy is (Gonzalez, 2008, p. 532)

$$S(Z) = -\sum_{z \in \Omega} p(z) \log_2(p(z)). \tag{2.5.5}$$

The treatment on first order texture has so far been abstract. The aim of the following discussion is to assess what these measures captures on an intuitive level. As noted before, first-order texture does not take into account the spatial organization of pixel intensities. Therefore, to understand first-order texture, it is sufficient to look at the histograms. Figure 2.5.2 shows some fundamental image histograms. Firstly, entropy S is the amount of disorder in the distribution. The more evenly the intensities are distributed over the bins, the higher the entropy. In figure 2.5.2, the extreme cases are shown in CONSTANT and FLAT, which illustrates perfect order and perfect disorder, respectively. In CONSTANT, all pixels fall into just one bin. That minimizes the entropy. In FLAT on the other hand, all intensity levels distribute equally over the bins, thus maximizing the entropy. Skewness $\hat{\alpha}_3$ can be understood as a measure of how intensity levels distribute about the mean. The more pixels on one side of the mean compared to the other side, the more skewness. SKEWED in figure 2.5.2 exemplifies a histogram of relatively great skewness.

[13]Note that the variance σ^2 is different from the variance s^2. σ^2 is called the n-normalized variance, because it is normalized by the number of samples, which correspond to the probability factors in equation (2.5.1). s^2 is called the *sample variance* or $n-1$-normalized variance because it is normalized by the number of samples minus one. For more information, see any good textbook on statistics, e.g. Alm and Britton (2008, p. 245).

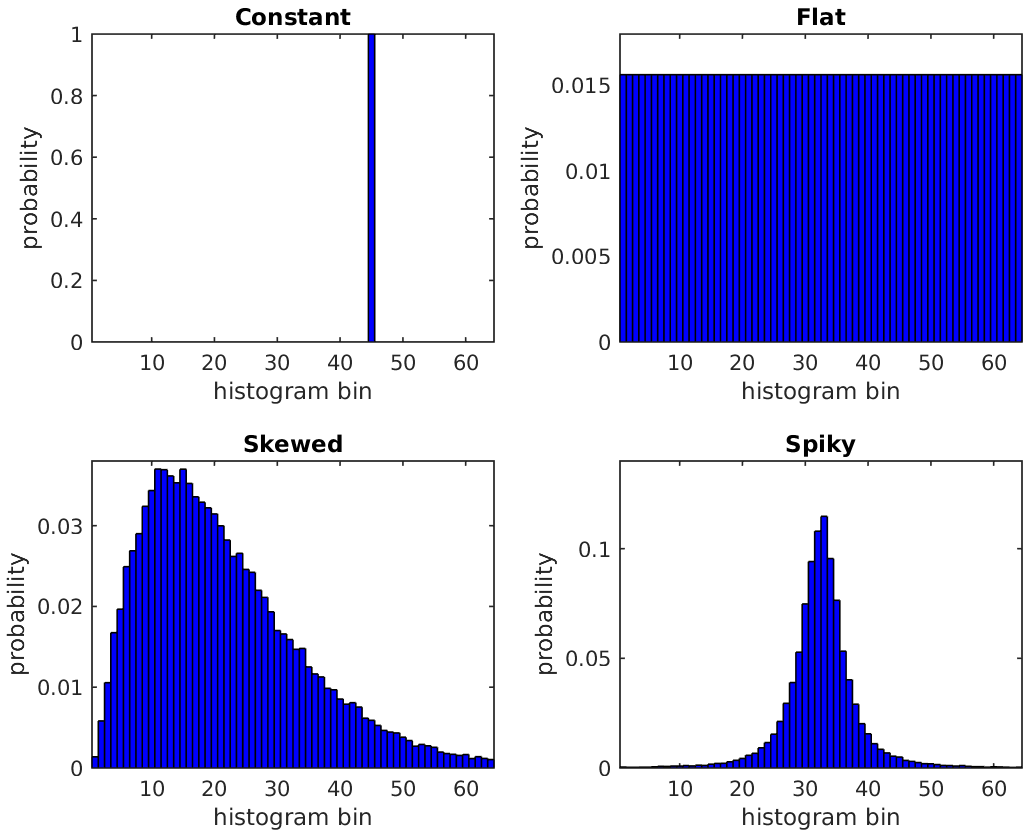


Figure 2.5.2: Some fundamental histogram shapes. In CONSTANT*, all pixels have the same intensity. In* FLAT*, all the intensity levels are equally common.* SKEWED *exemplifies a skewed histogram where pixel intensities tend to be darker than the average intensity. Finally, in* SPIKY*, a spike around the intensity level in the middle of the interval. All histograms have 64 bins. The original data had 256 intensity-levels.*

Finally, kurtosis $\hat{\alpha}_4$ can be understood as a measure of how spike-shaped the histogram is. SPIKY in figure 2.5.2 shows a histogram of relatively large kurtosis. The first order texture measures for the images behind the histograms in figure 2.5.2, and also for the imagery in figure 2.5.1, are composed in table 2.5.1. As expected the constant-intensity image CONSTANT, has the minimum entropy of zero, while the image with equally distributed intensities FLAT maximizes entropy. SKEWED displays the highest skewness and SPIKY displays the highest kurtosis. For the texture classes, SMOOTH and COARSE yield similar values, even though the texture content seems very different from inspection. It is however interesting that REGULAR has a high standard deviation and a low skewness. This is explained by the tendency of pixels in REGULAR to be either very bright, or very dark.

To conclude this section, first-order texture fails to separate between the fundamental texture classes in figure 2.5.1. The reason is that they don't take spatial information into account, i.e. are not spatial features. *Second-order textural features* on the other hand,

Table 2.5.1: Statistical measures for the imagery in 2.5.1 and the imagery behind the histograms in figure 2.5.2. The values was computed using all intensity-levels (256 bins). s denotes (n-1 normalized) sample standard deviation. $\hat{\alpha}_3$ denotes skewness and $\hat{\alpha}_4$ denotes kurtosis. S denotes entropy.

	s	$\hat{\alpha}_3$	$\hat{\alpha}_4$	S
SMOOTH	0.231	0.175	2.155	7.824
COARSE	0.121	0.130	3.132	6.985
REGULAR	0.355	0.035	1.493	7.001
CONSTANT	0.000	0.000	0.000	0.000
FLAT	0.290	0.000	1.800	8.000
SKEWED	0.193	0.865	3.395	7.507
SPIKY	0.092	0.010	7.986	6.389

not only considers what intensities occur, but also takes into account where they occur in relation to each other(Aggarwal and Agrawal, 2012). Second-order textural features are thus spatial features. A canonical approach to second-order textural features, utilizes the *Gray-level co-occurrence matrix* (GLCM) proposed by Haralick et al. (1973), (see also Gonzalez, 2008, p. 830-836). This is the topic of the next section.

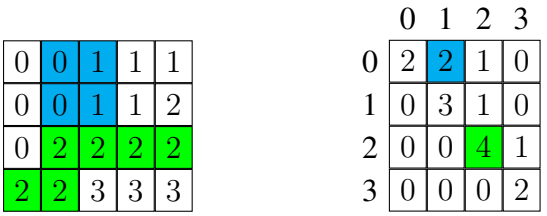
2.6 The Gray-level co-occurrence matrix (GLCM)

This treatment on the GLCM is based on (Gonzalez, 2008, p. 830-836). As briefly mentioned at the end of the previous section, the GLCM is a standard approach to texture classification. In principle, the GLCM is a two dimensional histogram that counts how frequently pairs of respective intensities (or grey-levels) co-occur. Co-occurrence is defined by a *spatial relationship operator* $\boldsymbol{q}$ that maps each *pixels of interest* to a neighbour pixel. For simplicity, $\boldsymbol{q}$ will be a fixed offset (although more complex spatial relationships are possible). The offset is represented by a vector. Initially, the *one-to-the-right-offset*, i.e. $\boldsymbol{q} = [0\ 1]^T$, is considered in the following examples. The GLCM G of image X corresponding to the spatial relationship $\boldsymbol{q}$ will be an $l \times l$ matrix. A particular value g_{ij} in G, is the count of intensity level i co-occurring with intensity level j (see figure 2.6.1 for an example). In practice, the image is usually *quantized* prior to construction, so that the size of G is manageable from a performance-perspective. In the examples that follows, eight distinct levels are utilized, which means that G is 8×8.

The original work by Haralick et al. (1973) utilizes a bi-directional GLCM, i.e. if the intensities (i, j) are related by the spatial relationship $\boldsymbol{q}$, then so are (j, i). Such a GLCM is obtained by adding G, with spatial relationship $\boldsymbol{q}$ to G', with spatial relationship $-\boldsymbol{q}$. However, changing the sign of the spatial relationship effectively swaps the co-occurrences from (i, j) to (j, i). Therefore, $G' = G^T$, and there is no need to compute more than one GLCM. In practice, the bi-directional GLCM G_2 is instead computed as

$$G_2 = G + G^T. \tag{2.6.1}$$

Like for image histograms, in order to take statistical measures directly from the GLCM, it must be a probability mass function (PMF), i.e. satisfy the two-variable counterparts



(a) Image *(b) GLCM*

Figure 2.6.1: A small sample image and its (one-directional, non-normalized) GLCM with the one-to-the-right spatial relationship a.k.a. $\boldsymbol{q} = [0 \quad 1]^T$*. The two colour labelled cells in the image and GLCM exemplifies two co-occurrences. Green correspond to co-occurrences of* $(0, 1)$*, and cyan to co-occurrences of* $(2, 2)$*.*

of equations (2.2.1), and (2.2.3). A PMF is obtained from dividing G_2 by the sum of its elements, i.e.

$$P = \frac{G_2}{\sum_{i=1}^{m}\sum_{j=1}^{n} G_2(i,j)}. \tag{2.6.2}$$

The bi-directional and normalized P will from here on be referred to just as the GLCM. All three steps to obtain P are illustrated in figure 2.6.2.

$$G = \begin{bmatrix} 2 & 2 & 1 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix} \qquad G_2 = \begin{bmatrix} 4 & 2 & 1 & 0 \\ 2 & 6 & 1 & 0 \\ 1 & 1 & 8 & 1 \\ 0 & 0 & 1 & 4 \end{bmatrix} \qquad P = \tfrac{1}{32}\begin{bmatrix} 4 & 2 & 1 & 0 \\ 2 & 6 & 1 & 0 \\ 1 & 1 & 8 & 1 \\ 0 & 0 & 1 & 4 \end{bmatrix}$$

(a) original *(b) bi-directional* *(c) normalized*

Figure 2.6.2: Steps for obtaining P*. The original* G *is the same as in figure 2.6.1. The bi-directional GLCM* G_2 *is symmetric because bi-directionality means that* $G_2(i, j) = G_2(j, i)$*. Also, the sum of all elements in* G_2 *is twice that* G *because both* (i, j) *and* (j, i) *is counted for each pixel-pair.* P *is both bi-directional and normalized (the sum of all elements equals* 1*). In effect,* P *contains the probabilities of co-occurrences. Strictly speaking, because* P *is bi-directional the probability of* (i, j) *co-occurring, is* $P(i, j) + P(j, i)$*, or equivalently* $2P(i, j)$*, if* $i \neq j$*. The probability of* i *to co-occur with itself is however just* $P(i, i)$*.*

2.6.1 GLCM features

The basest measures are the mean and variance. A possible extension to two-dimensional histograms is to define row-wise and column-wise means and variances. The row-wise mean is defined as

$$m_r = \sum_{(i,j)\in I\times I} ip_{ij}, \tag{2.6.3}$$

where I is the (possibly quantized) interval of intensity levels and $\times$ denotes the Cartesian product. Similarly, the column-wise mean is

$$m_c = \sum_{(i,j)\in I\times I} jp_{ij}. \tag{2.6.4}$$

The variances are defined as

$$\sigma_r^2 = \sum_{(i,j)\in I\times I} (i - m_r)^2 p_{ij} \tag{2.6.5}$$

and

$$\sigma_c^2 = \sum_{(i,j)\in I\times I} (j - m_c)^2 p_{ij}, \tag{2.6.6}$$

respectively. Further measures which are common in image analysis (Haralick et al., 1973), along with descriptions of their interpretations are summarized in table 2.6.1.

To get a better intuitive understanding of the features, the texture-class imagery in figure 2.5.1 and the chessboard imagery in figure 2.6.3 have been used to generate the results in table 2.6.2. The remainder of this section is a discussion of these results. The imagery SMOOTH, COARSE, and REGULAR will be referred to as the *texture set*, and the imagery CHESS1, CHESS2, and CHESS3 as the *chess set*. An important difference between these sets is that all imagery in the texture set displays some degree of smoothness, i.e. in a small neighbourhood the intensity increases gradually. In the Chess set on the other hand, the intensity between two adjacent pixels can go from 0 to 255 (eight-bit imagery is used).

All images except REGULAR display fairly equal values in the for horizontal and vertical features. In fact, all features seem to vary significantly with direction for REGULAR, which suggest that the GLCM features in multiple principal directions can describe regular texture. The results corresponding to SMOOTH and COARSE also show some results that are intuitive. The slow transition in SMOOTH yield a high correlation and homogeneity compared to COARSE. COARSE on the other hand has the higher contrast. A less intuitive result is that SMOOTH that certainly looks orderly, displays high entropy and low energy compared to COARSE.

Perfect vertical positive correlation (equal to 1) is according to table 2.6.2b seen in the image REGULAR. Because it consists only of columns of constant intensity, any pixel can only co-occur with another pixel of the same intensity. Thus, the GLCM is diagonal, which correspond to perfect correlation. The horizontal correlation of REGULAR is however not perfect. It is impossible to have both all columns and all rows constant without having the whole image constant. Also, as explained in table 2.6.1 correlation is undefined for zero GLCM-variances. If an image is constant, it has zero GLCM-variances.

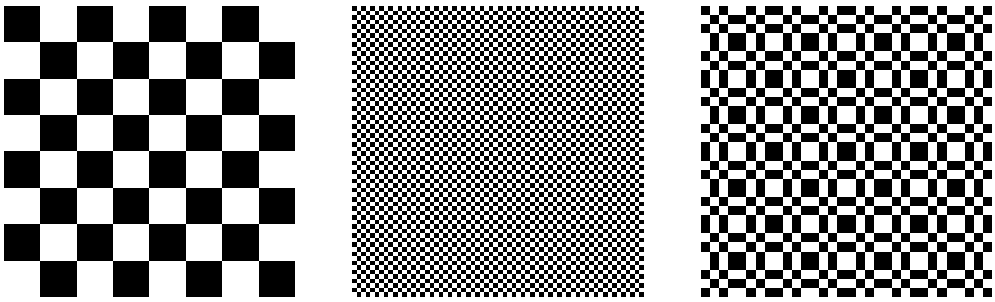
Table 2.6.1: Some of the most popular Haralick features derived from the normalized, bi-directional GLCM P. i and j are pixel intensities in the (possibly quantized) interval I.

Name	Interpretation	Formula
Maximum Probabillity	How often the most commonly co-occurring intensity levels co-occur.	$\max_{i,j}(p_{ij})$
Correlation	How correlated co-occurring pixels are, over the entire image. The measure takes values in the interval $[-1, 1]$. The measure is undefined if either σ_r^2 or σ_c^2 is zero.	$\sum_{i,j} \frac{(i-m_r)(j-m_c)p_{ij}}{\sigma_r^2\sigma_c^2}$
Contrast	Contrast of co-occurring intensities over the entire image. It takes values in $[0, (L-1)^2]$	$\sum_{i,j}(i-j)^2 p_{ij}$
Energy	Also known as *Angular Second Moment* (ASM), or *Uniformity*. Measures of the order in the image, and takes values in $[1/l^2, 1]$. It is 1 if the image is uniform and $1/l^2$ if the GLCM is single-valued.	$\sum_{i,j} p_{ij}^2$
Homogeneity	A measure of closeness to the GLCM diagonal in $[1/l, 1]$. It is 1 when the GLCM is diagonal (and intensities in the image only co-occur with themselves). It is $1/l$ when the GLCM is anti-diagonal.	$\sum_{i,j} \frac{p_{ij}}{1+\lvert i-j\rvert}$
Entropy	Is a measure of disorder in the GLCM with values in $[0, 2\log_2(l)]$. It is 0 when all elements in the GLCM except one are zero. It is maximized when all GLCM values are equal.	$-\sum_{i,j} p_{ij}\log_2(p_{ij})$

Therefore, perfect correlation is only possible in one principal direction. Perfect negative correlation is seen in CHESS2. The only non-zeros in its corresponding GLCM are the uppermost-rightmost and downmost-leftmost, respectively; which correspond to co-occurrences of $(0, 255)$. In general any anti-diagonal GLCM will have perfect negative correlation.

REGULAR and CHESS2 also displays the extremes for contrast. REGULAR has zero vertical contrast and CHESS2 has maximal contrast $49 = (l-1)^2$ in both principal directions. Homogeneity is maximized by REGULAR (vertically) and CONSTANT, and is minimized by CHESS2 to $0.125 = 1/l$.

CONSTANT has no texture component at all. It therefore takes extreme values for all features listed in the tables 2.6.2, except for correlation (for which it is undefined). This is the case for both principal directions. In particular, CONSTANT demonstrates perfect order, because it minimizes entropy and maximizes energy. To illustrate perfect GLCM disorder is quite difficult. A small example which maximizes entropy and minimizes energy is presented in figure 2.6.4.



(a) CHESS1 *(b)* CHESS2 *(c)* CHESS3

Figure 2.6.3: Three chessboard images of various frequencies, all of size 64×64*.* CHESS1 *is an ordinary* 8×8 *board, i.e. the length of a tile side or 'wavelength' is* 8 *pixels. In* CHESS2 *the wavelength is very short, just* 1 *pixel. The last board* CHESS3 *consists of the xor (exclusive or) of the ordinary chessboard* CHESS1 *and a high-frequency chessboard with wavelength* 2 *pixels. Thus, there should be two dominating frequencies in* CHESS3*.*

Table 2.6.2: GLCM features for horizontal offsets $\boldsymbol{q} = [0\ 1]^T$*, and vertical offsets* $\boldsymbol{q} = [1\ 0]^T$*. The features are calculated for the texture-class imagery in figure 2.5.1, and for the chess imagery in figure 2.6.3. Because* CONSTANT *has zero GLCM-variance correlation is undefined for the particular image (see table 2.6.1).*

(a) Horizontal offset

	Max(P)	**Correlation**	**Contrast**	**Energy**	**Homogeneity**	**Entropy**
SMOOTH	0.177	0.997	0.019	0.140	0.991	3.011
COARSE	0.341	0.927	0.149	0.211	0.926	2.785
REGULAR	0.212	0.979	0.314	0.108	0.843	3.851
CONSTANT	1.000	-	0.000	1.000	1.000	0.000
CHESS1	0.444	0.778	5.444	0.401	0.903	1.503
CHESS2	0.500	−1.000	49.000	0.500	0.125	1.000
CHESS3	0.310	0.238	18.667	0.264	0.667	1.959

(b) Vertical offset

	Max(P)	**Correlation**	**Contrast**	**Energy**	**Homogeneity**	**Entropy**
SMOOTH	0.177	0.997	0.019	0.140	0.991	3.011
COARSE	0.340	0.926	0.150	0.209	0.925	2.790
REGULAR	0.234	1.000	0.000	0.157	1.000	2.838
CONSTANT	1.000	-	0.000	1.000	1.000	0.000
CHESS1	0.444	0.778	5.444	0.401	0.903	1.503
CHESS2	0.500	−1.000	49.000	0.500	0.125	1.000
CHESS3	0.310	0.238	18.667	0.264	0.667	1.959

0	1	1
1	0	0

(a) Image

	0	1
0	1	1
1	1	1

(b) GLCM

Figure 2.6.4: A small binary image of perfect GLCM disorder (using one-to-the-right spatial relationship). Its Entropy is $2 = 2\log_2(2)$*, and its Energy is* $1/4$*. Note that the the GLCM is not bi-directional, neither is it normalized.*

2.6.2 GLCM at multiple scales

So far we have only considered step lengths of length 1. In practical applications, this step length might bee to small to reveal interesting detail. This section investigates what information can be revealed from looking at the texture set imagery at multiple scales.

Intuitively, smooth texture should be characterized as having little detail at fine scales. This can also be seen for SMOOTH in figure 2.6.5, because at a finer scale the contrast is low, but the correlation is high. However, because the image SMOOTH display a systematic change from one side of the image to the other, the contrast increases with the scale.

The COARSE image displays no systematic intensity change. Hence, the features flattens out when the step length gets sufficiently large as seen in figure 2.6.6. Interestingly, at the finest scales (step lengths near 1), the features of COARSE takes values similar to SMOOTH.

Finally, the multiscale plot of the REGULAR image clearly shows frequency information as seen in figure 2.6.7. Both Correlation and Contrast becomes sinusoids with the same frequency as the pattern in REGULAR. The other features are also periodic, but correspond to more complex functions of the step-length.

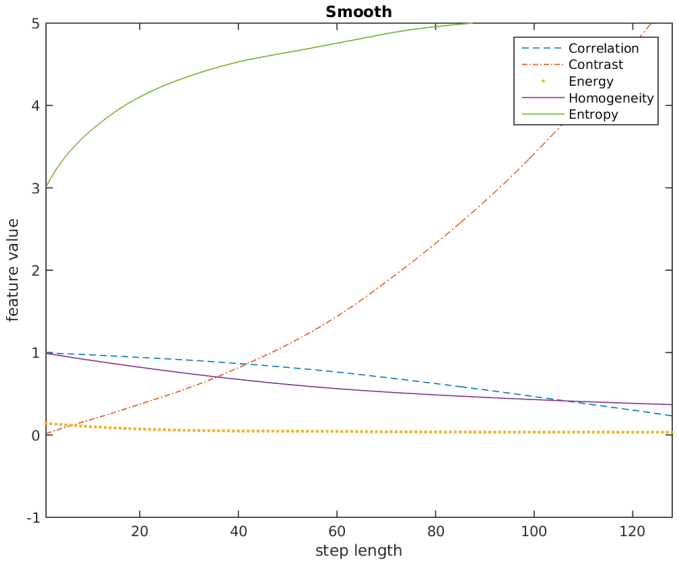


Figure 2.6.5: GLCM features of SMOOTH*, at different scales. Step lengths from 1 to 128 was used, and the GLCM has eight levels. Because the image displays systematic change, the contrast increases smoothly with the step-length. Also, note that the the plots corresponding to correlation and homogeneity almost coincide.*

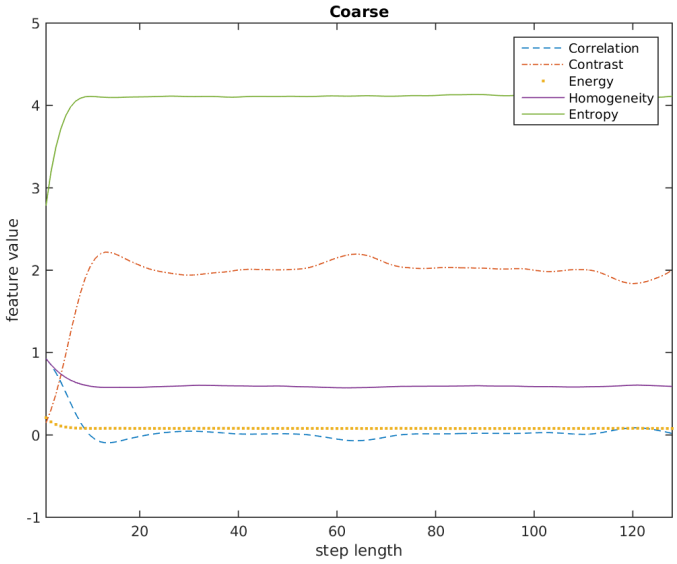


Figure 2.6.6: GLCM features of COARSE*, at different scales (see figure 2.6.5). For small step lengths, all features develops rapidly up a step-length of about* 16*, but then flattens out. This tells us that the details in the* COARSE *are smaller than 16 pixels. Also, the flat tail means that the image has no systematic change.*

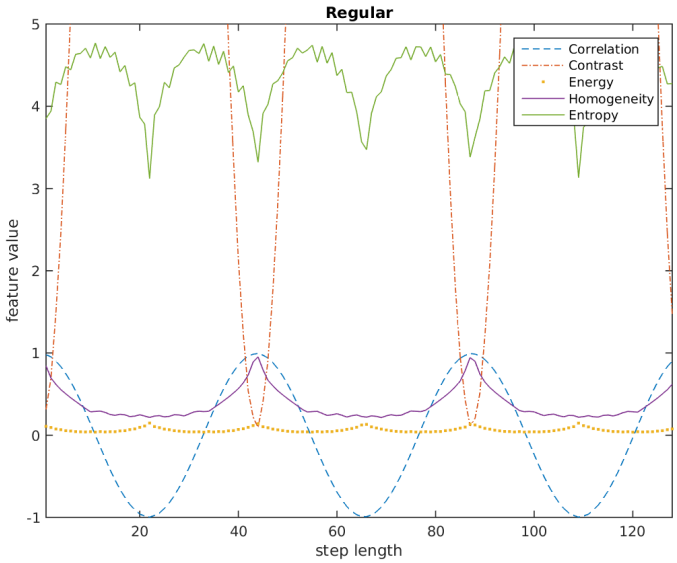


Figure 2.6.7: GLCM features of REGULAR, *at different scales (see figure 2.6.5). All features display distinguishable periodic pattern with a period of* 42.7 *pixels. The correlation and homogeneity features even form sinusoidal patterns.*

2.7 Wavelets and multiresolution theory

The difficulty of choosing the appropriate scale to look for texture, or spatial features in general, has already been noted (see section 1.2). The remedy to this fundamental problem is to look at multiple scales at once. In signal processing, this is simple idea has spawned the entire sub-field of *multiresolution theory*, and its foundation is the *wavelet transform* (Mallat, 1989). Because images can be viewed as two-dimensional signals, wavelets are also applicable to image analysis. Wavelets are small waves of specific frequencies and limited duration (Gonzalez, 2008, p. 461). The latter property is of particular importance in image analysis, since images are always of limited duration. This brief introduction to wavelets will focus on the *Haar wavelet*. A final note before we start is that the signals considered in the remainder of this introduction to wavelets will have sizes $n = 2^k$ for $k \in \mathbb{N} \cup \{0\}$. That is, because we want to be able to sub-sample a digital signal until its size equals one.

2.7.1 The Haar wavelet

The first and simplest wavelet was suggested in Haar (1910). It is therefore known as the *Haar wavelet*. The explanation given in this section is based on (Strang, 2009, p. 391-392) and (Gonzalez, 2008, chap. 7).

Starting with the one-dimensional case, a digital signal can for our purposes be considered a real vector $\boldsymbol{x}$ (here we will use column vectors). Furthermore, the length of x is limited to powers of twos, i.e. $n = 2^k$ for $k \in \mathbb{N} \cup \{0\}$. The *Haar transform* is then a recursive function which takes averages and differences of adjacent elements in $\boldsymbol{x}$, and in each step sub-samples it (i.e. reduces the number of elements). See algorithms 1 and 2.

In the algorithms, the vector $\boldsymbol{a}$ is called an *approximation* of $\boldsymbol{x}$ and the vector $\boldsymbol{d}$ the *detail* of $\boldsymbol{x}$. The output of algorithm 1 is the vector of *haar coefficients* $\boldsymbol{c}$, that has the format

$$\boldsymbol{c} = \begin{bmatrix} a \\ \boldsymbol{d}_k \\ \boldsymbol{d}_{k-1} \\ \vdots \\ \boldsymbol{d}_1 \end{bmatrix}, \tag{2.7.1}$$

where $k = \log_2(n)$ is the depth of recursion, and a is the scalar approximation of the signal, i.e. the signal approximated as a single value. The decomposition procedure is illustrated in figure 2.7.1.

Each recursion of algorithm 1 effectively halves the resolution in the approximation $\boldsymbol{a}$ and extracts detail components $\boldsymbol{d}$ of doubling coarseness. Specifically, the sizes of $\boldsymbol{a}_{i+1}, \boldsymbol{d}_{i+1}$ is half that of $\boldsymbol{a}_{i+1}, \boldsymbol{d}_{i+1}$, respectively. In particular, this means that the *wavelength* of details in $\boldsymbol{d}_i$ at any level i, will be $2i$. In this sense $\boldsymbol{c}$ represents the image on multiple scales. Furthermore, $\boldsymbol{c}$ is the same size as the original signal $\boldsymbol{x}$, meaning that it is a compact representation of $\boldsymbol{x}$. Also, the original signal x can be perfectly recovered from $\boldsymbol{c}$ by applying the *inverse Haar transform*. Omitting the details, the inverse transform

function *haar-transform(**x**)***begin***
  $(\boldsymbol{a}, \boldsymbol{d}) :=$ *haar-step(**x**)*
  if $length(\boldsymbol{a}) > 1$ **then**
    $\boldsymbol{c}_{head}$:= *haar-transform($\boldsymbol{a}, \boldsymbol{d}$)*
  else
    $\boldsymbol{c}_{head} := \boldsymbol{a}$
  end
  return: $\begin{bmatrix} \boldsymbol{c}_{head} \\ \boldsymbol{d} \end{bmatrix}$
end

Algorithm 1: One dimensional Haar-transform.

function *haar-step(**x**)***begin***
  for *each odd i from 1 to length(**x**)-1* **do**
    $a_i := \frac{x_{2i}+x_{2i+1}}{2}$
    $d_i := \frac{x_{2i}-x_{2i+1}}{2}$
  end
  return: $(\boldsymbol{a}, \boldsymbol{d})$
end

Algorithm 2: Calculation of the Haar wavelet approximation- and detail- components.

up-samples the approximations by adding the detail components back, that is

$$\boldsymbol{a}_{k-1}(2i) = \boldsymbol{a}_k(i) - \boldsymbol{d}_k(i) \tag{2.7.2}$$

and

$$\boldsymbol{a}_{k-1}(2i+1) = \boldsymbol{a}_k(i) + \boldsymbol{d}_k(i), \tag{2.7.3}$$

until $k = 0$ (note that the zero-level approximation is the original signal $\boldsymbol{a}_0 = \boldsymbol{x}$). Finally, it is not necessary to decompose the signal to one value. In practice, the recursion of algorithm 1 can be truncated at a specific level of depth that befits the application.

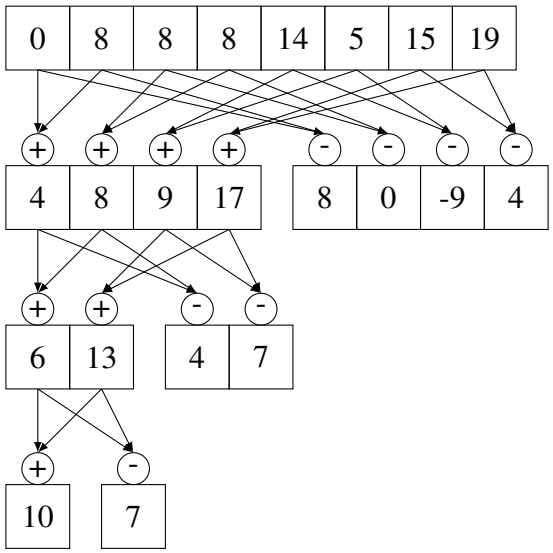


Figure 2.7.1: Illustration of the 1D Haar transform. The leaf-nodes form the coefficients $\mathbf{c}$ *in the correct order. Figure provided by Niclas Börlin. Used with permission.*

2.7.2 Extension of the Haar-transform to 2D

To extend the *haar-step* to the two-dimensional case is a simple matter of applying the one dimensional Haar-transform to each row, and then to each column of X, resulting in an approximation A and three detail components D_H, D_V, and D_D, each of which is $1/4$ the size of X (area-wise). The details are as follows. The transform *haar-step-columns* is defined to take a matrix for input and apply *haar-step* to each column of the matrix. That is, if $X = \left[\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_n\right]$, where $\boldsymbol{x}_j$'s are columns, then

$$\textit{haar-step-column}(X) = \left[\textit{haar-step}(\boldsymbol{x}_1) \quad \textit{haar-step}(\boldsymbol{x}_2) \quad \ldots \quad \textit{haar-step}(\boldsymbol{x}_n)\right]. \tag{2.7.4}$$

Similarily,

$$\textit{haar-step-row}(X) = \begin{bmatrix} \textit{haar-step}(\boldsymbol{y}_1)^T \\ \textit{haar-step}(\boldsymbol{y}_2)^T \\ \vdots \\ \textit{haar-step}(\boldsymbol{y}_m)^T \end{bmatrix}, \tag{2.7.5}$$

where the $\boldsymbol{y}_i$'s are columns of X. The definition of the 2D transform is trivial:

$$\textit{haar-step-2D}(X) = \textit{haar-step-rows}(\textit{haar-step-columns}(X)). \tag{2.7.6}$$

The output matrix will have the block structure

$$\textit{haar-step-2D}(X) = \begin{bmatrix} A & D_H \\ D_V & D_D \end{bmatrix}, \tag{2.7.7}$$

where A is called the approximation, D_H the horizontal detail, D_V the vertical detail, and D_D the diagonal detail. As in the 1D case, the two dimensional *haar-transform* is obtained by recursive application of *haar-step-2D* on successive approximations. A limited depth 2D *haar-transformation* is illustrated in figure 2.7.2.

(a) Haar decomposition

$$\begin{bmatrix} \begin{bmatrix} A_2 & D_{H2} \\ D_{V2} & D_{D2} \end{bmatrix} & D_{H1} \\ D_{V1} & D_{D1} \end{bmatrix}$$

(b) Block structure

Figure 2.7.2: Example of the two dimensional haar transform (with recursion depth limited to two levels). (a) shows a sample Haar decomposition of a Passion flower image. (b) shows the decompositions block structure. In the decomposition there are six detail components corresponding to two different scales and the three different principal directions. Also, the second-level approximation is seen in the upper-left corner. In particular, note how the detail components captures the flower petals. In D_{H1} and D_{H2} the vertically aligned petals are more distinctive. Correspondingly, in D_{V1} and D_{V2} the horizontally aligned petals are more distinctive. D_{D1} and D_{D2} both display a cross-pattern. Passion flower image provided by Pixbay, `http://pixabay.com`.

2.8 Supervised learning and classification

Machine learning is about estimating some *unknown function* f by another function h. h is called an *hypothesis*. In *supervised learning*, the value of f is known at a finite set of points from which the h is built. Formally, a supervised-learning problem is recognized as follows: Given a *training set* of m *input-output pairs* $X_{train} = \{(\boldsymbol{x}_1, y_1), (x_2, y_2), ...(\boldsymbol{x}_m, y_m)\}$, such that $y_i = f(\boldsymbol{x}_i)$ for all $1 \leq i \leq m$, find h that approximates f.

Supervised learning problems where the the possible values of f are a finite set are called *classification problems*. For classifiers, the input $\boldsymbol{x}_i$'s are called *observations* and the output y_i's are called *class-labels*. Observations are typically so-called *feature-vectors*, i.e. vectors of n *features*[14]. For convenience the observations are organized into a *feature matrix*, where rows correspond to observations and columns to features. Also, the class labels are organized corresponding to observations, i.e.

$$X = \begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \vdots \\ \boldsymbol{x}_m^T \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}. \tag{2.8.1}$$

The hypothesis, when given a feature matrix will return a vector of *predictions* corresponding to each observation, i.e.

$$\boldsymbol{y}^* = h(X). \tag{2.8.2}$$

For more information on supervised learning and classification, see Russell and Norvig (2010, chapter 18).

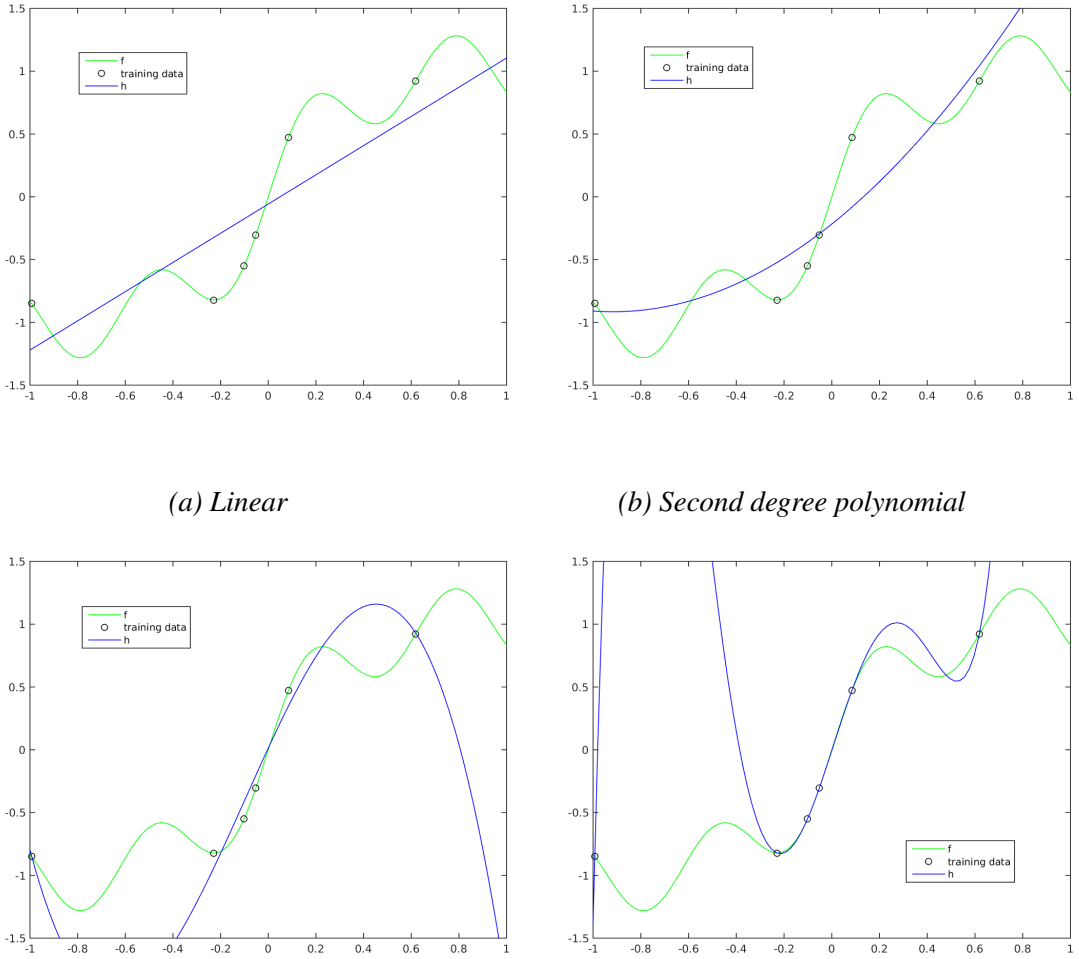
[14] In some literature, the name *attributes* is used instead of features.

2.8.1 Generalization and overfitting

For clarity, this section deals with continuous y_i's, i.e. the problems in the examples are not classification problems. But the conclusions apply to classification problems as well. Let h denote a hypothesis, trained with the *training data* $X_{train}, \boldsymbol{y}_{train}$. The training data was sampled from the function f. The fraction of correctly classified observations in X_{train} will differ, depending on what kind of h is chosen and of f. If all observations are correctly classified, i.e. if $y_{train} = y^*_{train} = h(X_{train})$, then h is called a *consistent hypothesis*. In many cases, the degree of consistency (fraction of correctly classified training data) can be adjusted by varying the complexity of h. For example, if h is a polynomial, the degree of the polynomial can model complexity. However, the usefulness of a hypothesis is not determined by how well it can describe the training data, because we already know the real values $\boldsymbol{y}_{train}$. The real usefulness is determined by how well h can classify new data. A h that does well on new data is said to *generalize* well. That is, if $X_{test}, \boldsymbol{y}_{test}$ denotes the *test data*, that like the training data is sampled from f, but not used to train h. Then, h trained with $X_{train}, \boldsymbol{y}_{train}$ generalizes well if $h(X_{test}) = \boldsymbol{y}^*_{test} \approx \boldsymbol{y}_{test}$.

The issue of *overfitting* is that consistency of h does not guarantee that h generalizes well. On the contrary, there is often a simplicity-complexity trade-off between simple h's which may generalize better, and complex h's that describes the data in more detail. This trade-off is illustrated in figure 2.8.1, for polynomial h. Intuitively, overfitting can be illustrated by a student which memorizes the answers to all exercise questions, rather than to understand them. Overfitting can happen for all types of learners, even when h is constrained to be very simple. However, in general, the influence of overfitting increases with the number of features, but decreases when more observations are added to the training data. In figure 2.8.2 the complex type of hypothesis from figure 2.8.1d is remedied by the addition of training data.

This section is based on (Russell and Norvig, 2010, chap. 18.3.5).



(a) Linear

(b) Second degree polynomial

(c) Third degree polynomial

(d) Fifth degree polynomial

Figure 2.8.1: Illustration of overfitting with a training set of five observation-value pairs. The hypotheses h is a polynomial of varying complexity. The two simplest hypothesis in a and b) generalizes better than the more complex hypotheses used in c and d), because of the fact that the simpler hypotheses comes closer to the real function f at points that are not very near the training data. The only consistent hypothesis is seen in d. It generalizes poorly because it diverges rapidly from f outside the interval containing the train data.

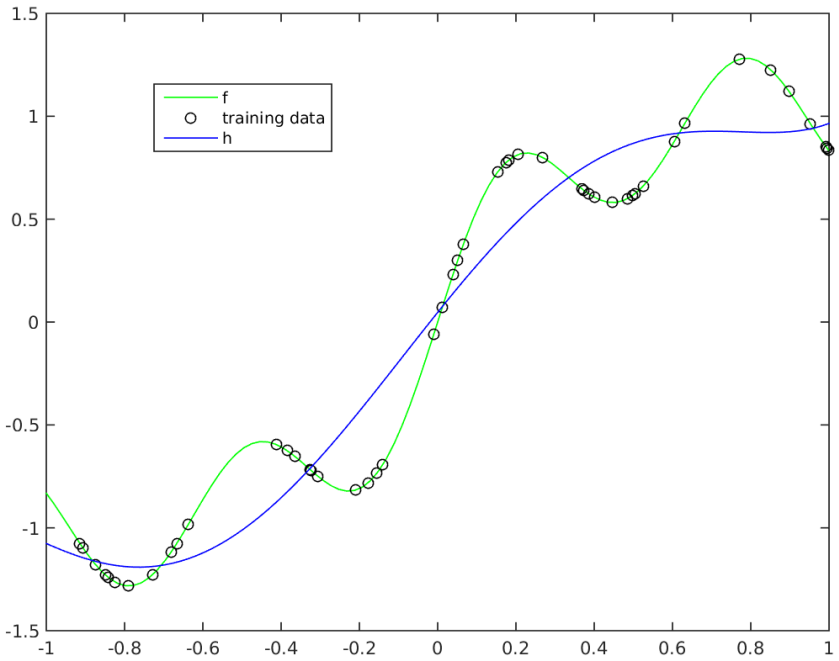


Figure 2.8.2: If the set of training data is increased to 50 *pairs, the overfitting problem diminishes for the fifth degree polynomial hypothesis.*

2.8.2 Classification performance

The previous section stated that a good hypothesis generalizes well, i.e. performs well on test data. This section quantifies how to measure generalization. In the remainder of the article, the term *classification performance* will be used synonymously with classifier generalization.

In general, the classification performance can be evaluated if the cost of misclassification is known (which is very domain specific). That is, the loss of classifying $h(\boldsymbol{x}) = \boldsymbol{y}^*$ is given by a function $L(y, y^*)$[15] (note that scalar class labels y, y^* are considered in the definition of the function). Also, let E denote the set of all possible observation-, class-label- pairs, i.e. all pairs $(\boldsymbol{x}, y)$, for any $\boldsymbol{x}$ in the feature-vector domain. Also, let $P(\boldsymbol{x}, \boldsymbol{y})$ denote the prior probability distribution that the feature vector $\boldsymbol{x}$ occurs with the class data $\boldsymbol{y}$. The *Generalization loss* is then given by

$$GenLoss_L(h) = \sum_{(\boldsymbol{x}_i, y_i) \in E} L(y_i, h(\boldsymbol{x}_i)) P(\boldsymbol{x}_i, y_i). \tag{2.8.3}$$

This general measure denotes the real performance of the classifier if L is properly set[16]. However, in most practical problems neither E nor P are known. If instead all available data E^*[17] is used as an estimation of E, an estimation of the generalization loss called *empirical loss* is given by

$$EmpLoss(h) = \sum_{(\boldsymbol{x}_i, y_i) \in E^*} L(y_i, h(\boldsymbol{x}_i)). \tag{2.8.4}$$

Finally, to make the result easier to interpret, it is sometimes useful to normalize the loss by dividing it by the number of observations m. The normalized empirical loss is then

$$NEmpLoss(h) = \frac{1}{m} \sum_{(\boldsymbol{x}_i, y_i) \in E^*} L(y_i, h(\boldsymbol{x}_i)). \tag{2.8.5}$$

A simple loss function is

$$L_{\frac{0}{1}}(y, y^*) = \begin{cases} 0 & \text{if } y = y^* \\ 1 & \text{otherwise} \end{cases}. \tag{2.8.6}$$

The normalized empirical loss using $L_{\frac{0}{1}}$ is the fraction off misclassifications

$$NEmpLoss_{L_{\frac{0}{1}}}(h) = \frac{n_{wrong}}{n_{right} + n_{wrong}}, \tag{2.8.7}$$

where n_{right} and n_{wrong} refers to the number of correct and incorrect predictions, respectively.

[15] In general $L(x, y, y^*)$, but for this study the simpler version that omits $\boldsymbol{x}$ is sufficient.

[16] At least for the more general version.

[17] Technically speaking, E^* is not a set, because it is possible for distinct observations to have the same class labels and feature vector. One way to deal with this is to let E^* denote an ordered array of feature-vector-, class-label- pairs.

This measure is suitable when the cost of misclassification is similar for all classes, and when the class frequencies are the same. But, when the class frequencies are different, $NEmpLoss_{L_{\frac{0}{1}}}$ is problematic. As an example, consider a forest of 999 spruces and just one pine. Let the hypothesis h be, 'all trees in the forest are spruces,' i.e. $h(\boldsymbol{x}) = spruce$. Then $NEmpLoss_{L_{\frac{0}{1}}}(h) = .001$ which is considerably small. Does that mean that h was a good hypothesis? When all classes in the population are equally important, but the fraction of classes in the population are different, it is feasible to use the *class-normalized empirical loss*. This loss assigns equal importance to all classes. This means that all classes of any kind contributes with $1/k$ to $NEmpLoss$. For example, to wrongly classify all observations of class i, but to correctly classify all instances of the $k-1$ other classes will yield a normalized empirical loss of $1/k$. The class-normalized loss function is defined as

$$L_{cn}(y_i, y^*) = \begin{cases} 0 & \text{if } y_i = y^* \\ \frac{m}{k^2 n(y)} & \text{otherwise,} \end{cases} \tag{2.8.8}$$

where k is the number of classes, and $n(y)$ is the count of observations of the class y. The normalized empirical loss using L_{cn} can be expressed:

$$NEmploss_{L_{cn}}(h) = \frac{1}{k} \sum_{i=1}^{k} \frac{n_{wrong}(i)}{n(i)}, \tag{2.8.9}$$

where $n_{wrong}(i)$ denotes the number of wrong predictions of the class i.

This section is based on (Russell and Norvig, 2010, chapter 18.4.2).

2.8.3 Cross-validation

As explained in section 2.8.1, when evaluating performance the training set and test set must be disjoint. This means that the feature matrix X and class labels $\boldsymbol{y}$ corresponding to the whole dataset, must be partitioned into training data X_{train}, $\boldsymbol{y}_{train}$ and test data X_{test}, $\boldsymbol{y}_{test}$. The obvious problem with this is that we cannot fully utilize the whole dataset. To produce a reliable classifier we want to train it with as much test data as possible. Also, we want to use as much data in the training set as possible, in order to produce statistically reliable performance result.

This problem is solved by a method called k*-fold cross-validation*, that works as follows. First, divide the data into k equally sized partitions: $X_1, X_2, ...X_k$, $y_1, y_2, ...y_k$. Evaluate the result for k folds. In each fold, take all except one of the partitions as training data, and use the single remaining partition for test data. That is, for the i'th fold let

$$X_{train} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_{i-1} \\ X_{i+1} \\ \vdots \\ X_k \end{bmatrix}, \boldsymbol{y}_{train} = \begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \\ \vdots \\ \boldsymbol{y}_{i-1} \\ \boldsymbol{y}_{i+1} \\ \vdots \\ \boldsymbol{y}_k \end{bmatrix}$$

and let

$$X_{test} = X_i, \boldsymbol{y}_{test} = y_i.$$

5 and 10 are popular values for k (Russell and Norvig, 2010, chapter 18.4.0).

In each fold, a performance metric such as that in equation (2.8.7) is computed. Alternatively, the predictions generated in each fold can be stored to generate a *compound prediction*, and performance evaluated on all predictions. In particular, the class-normalized loss in (2.8.9) computed on all predictions is likely to yield a different result than computing it in each fold and taking the average of the folds. The reason is that in the former case $n(i)$ and $n_{wrong}(i)$ are based on all data. In the later case they are just based on test data.

2.9 The support vector machine (SVM)

The SVM is today the most popular "off-the-shelf" classifier, because it is simple, computationally efficient, does not depend on parameters in its simplest form, and because it performs well in many cases (Russell and Norvig, 2010, p. 744). In this introduction, only linear SVM's will be considered, but SVM's are extendable to non-linear cases cases as well.

The SVM is a *binary classifier*, which means that it can only discriminate two classes. The classes are traditionally referred to as the positive- and negative- class, or alternatively the $+1$ class and the -1 class. Basically, the SVM constructs a *decision boundary* that best separates the two classes in the training data. Specifically, the boundary is chosen as the hyperplane that maximizes the margin between the *nearest* observations of respective classes, i.e. a *maximum margin separator*. In order to predict a test observation, the SVM uses a *decision function* d that is a signed distance measure from the observation (feature vector) to the decision boundary. If the sign is positive, the observation is predicted to belong to the $+1$ class, and vice versa for the -1 class (hence the names). The hypothesis function h corresponding to the SVM is given by

$$h(\boldsymbol{x}) = sign(d(\boldsymbol{x})), \tag{2.9.1}$$

where additionally the case of $d(\boldsymbol{x}) = 0$ may have to be defined in practice. Figure 2.9.1 illustrates the SVM. For more information on SVM's see also Burges (1998).

2.9.1 Extension to more than two classes

For multi class problems involving more than two classes, several binary SVMs can be combined in order to produce a multi-class classifier, or multi-SVM. Two simple, canonical approaches is the *one-versus-all method* and the *one-versus-one* method (Hsu and Lin, 2002).

The one-versus-all method trains k partial-SVMs, each corresponding to one class. The partial-SVM corresponding to the class i is trained with observations of class i with the positive labels and observations of other classes with negative labels. Thus, all observations are used in each classifier. Prediction of a test data sample is done by choosing

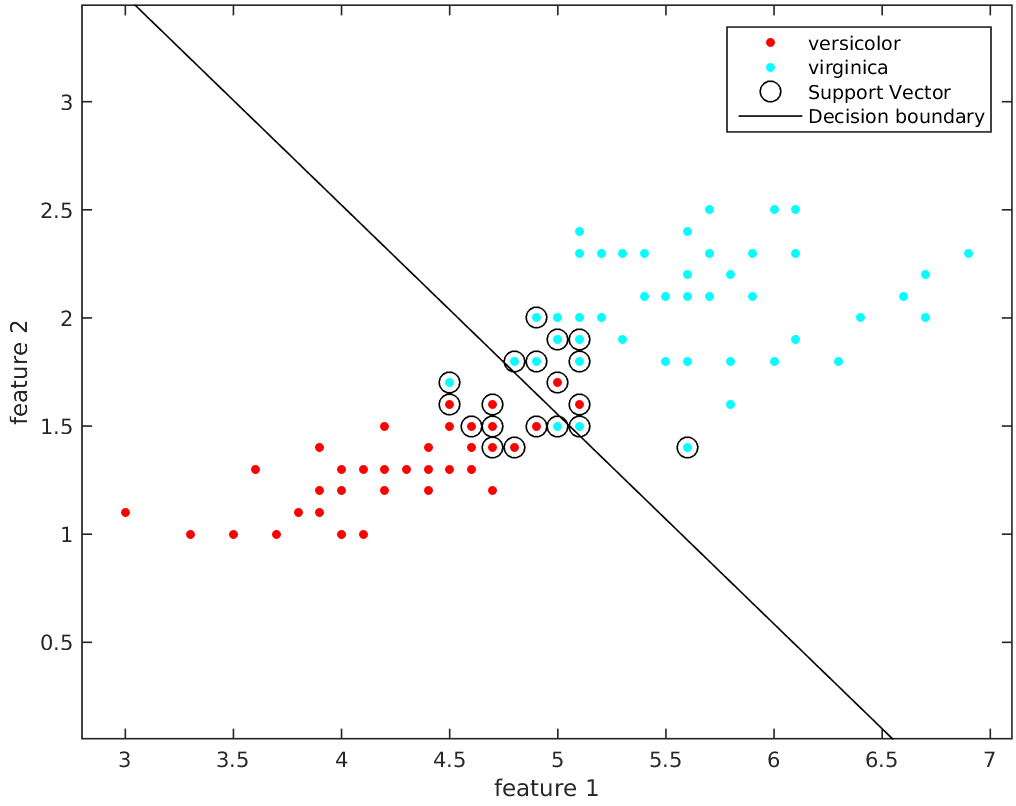


Figure 2.9.1: Illustration of a SVM feature space and the SVM decision boundary. In this case, the SVM is not consistent because it fails to classify some of the training data. This means that the the maximum margin is negative. Only the support vectors need to be used in computations, which is one of the reasons for SVMs efficiency. This example utilizes the iris data in Fisher (1936). Versicolor and Virginica are the names of the classes.

the class corresponding to the partial-SVM that maximizes the decision function (d in the previous section) (Hsu and Lin, 2002).

The one-versus-one method trains $k(k-1)/2$ partial-SVMs corresponding to pairs of classes. Specifically, the partial-SVM corresponding to the pair (i, j) is trained using only the observations of class i (with negative labels) and class j (with positive labels). Once the partial-SVMs have been trained, prediction can be done with this voting scheme: Let $\boldsymbol{x}$ denote the feature vector corresponding to the observation to predict. Also, let v denote a vector of length k (one cell per class) that is initially set to zero. For each partial-SVM: predict $\boldsymbol{x}$. If predicted as -1, then increment $v(i)$ by one. Otherwise increment $v(j)$ by one. When this is done, v is a vector of votes. The class receiving the most votes is chosen as the compound prediction. Alternatively, on a tie, the prediction is either marked as *unresolved*, or the prediction has to be determined by some other method (Hsu and Lin, 2002).

2.10 Feature selection and feature ranking for classification problems

For real-world classification problems in general, little is known about how features relate to classes. In particular, a features *relevance* and *redundancy* are important concepts that are scarcely understood. Relevance is a particular features ability to discriminate between given classes. Redundancy, or rather non-redundancy, refers to the amount of classification performance the particular feature adds over the other features used in classification. A canonical method for dealing with the lack-of-knowledge problem is to introduce a multitude of candidate features. The drawback of this approach is that irrelevant and redundant information is added, which causes overfitting. The classifier performance could be improved, if it was possible to keep only relevant, non-redundant features (Tang et al., 2014).

The problem of *feature selection* consist of finding an optimal subset of features, e.g. optimal in the sense of classification performance, or runtime. Given a particular classifier, an exhaustive search for the optimal subset of features can be done by evaluating performance for all $2^n - 1$ subsets of features. This leads to combinatorial explosion. Therefore, in practice for large feature sets, we have to be content with a sub-optimal subset (Tang et al., 2014).

A distinct but closely related problem is *feature ranking*. Feature ranking assigns a rank to each feature, in the form of a unique number ranging from 1 to m. The result is a top m list of features. Feature ranking is easily transformed to feature selection, e.g. by keeping the top m' features, or by repeatedly removing the worst feature until a loss in classification performance supersedes a threshold value (Guyon et al., 2002).

2.10.1 Feature ranking with the SVM

The method described here was suggested in (Guyon et al., 2002). It was originally developed for linear SVMs, but is general to classifiers that are *linear* and *multivariate* (optimized during training to handle multiple features). The basic idea of the algorithm is that the the more orthogonal the decision boundary is to the feature, the more important the feature is. This is illustrated in figure 2.10.1. The method takes an approach to feature ranking called *recursive feature elimination*. At every step, the SVM is trained using features that are not ranked (initially all). The feature that is most parallel to the decision boundary is assigned a rank, and thus is excluded from successive steps. Algorithm 3 provides the details.

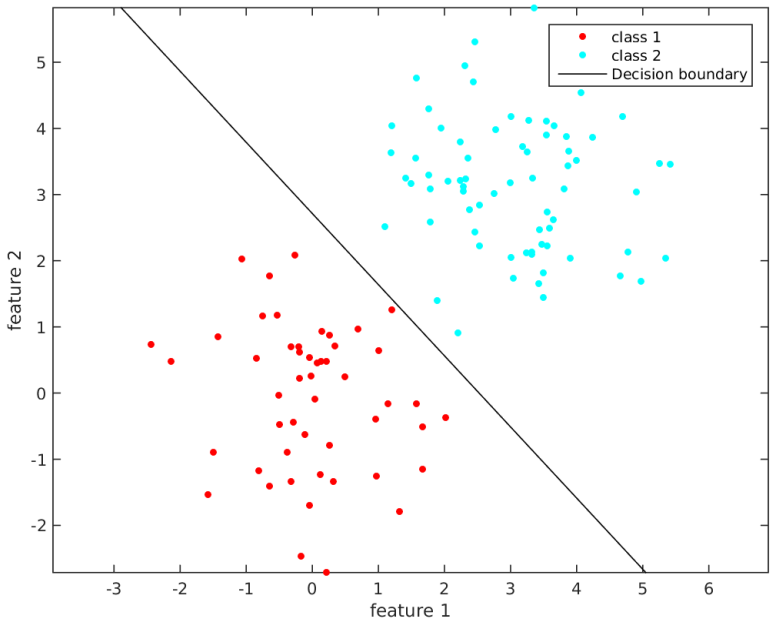
function *svm-rfe(*$X_0, \boldsymbol{y}$*)***begin**
  $\boldsymbol{s} := \begin{bmatrix}1 & 2 & \dots & n\end{bmatrix}$ # The set of surviving features.
  $\boldsymbol{r} := [\,]$ # Set of ranks.
  while $s \neq [\,]$ **do**
    # Get the columns of X_0 corresponding to surviving features.
    $X := X_0(:, \boldsymbol{s})$
    $h := trainSVM(X, \boldsymbol{y})$
    # Obtain the normal to the decision boundary.
    $\boldsymbol{w} := decisionBoundaryNormal(h)$
    # $\hat{*}$ denotes the element wise products.
    # The multiplication removes the sign.
    $\boldsymbol{c} := \boldsymbol{w} \hat{*} \boldsymbol{w}$
    # Index least important feature that still survives.
    $f := indexOfMin(\boldsymbol{c})$
    # Add the feature to the rankings and remove it from surviving features.
    $\boldsymbol{r} := \begin{bmatrix}s_f & \boldsymbol{r}\end{bmatrix}$
    $\boldsymbol{s} := \begin{bmatrix}\boldsymbol{s}(1 : f-1) & \boldsymbol{s}(f+1 : length(s))\end{bmatrix}$
  end
  return: **r**
end

Algorithm 3: Feature ranking algorithm. MatLab notation is used for readability.

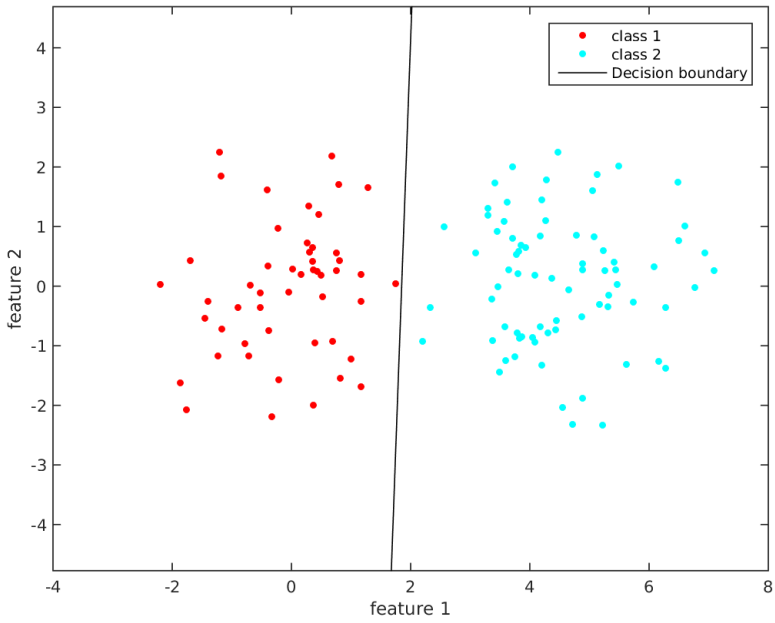
A problem with this approach is that the selection is subject to scale bias, as illustrated in figure 2.10.2. To deal with this, Guyon et al. (2002) suggests using a simple feature scaling strategy. Substract the mean m from each feature and divide the result by the standard deviation s, i.e.

$$\bar{\boldsymbol{f}} = \frac{\boldsymbol{f} - m(\boldsymbol{f})}{s(\boldsymbol{f})}, \tag{2.10.1}$$

where $\boldsymbol{f}$ denotes a vector of some particular feature for all observations, i.e. a column in the feature matrix.



(a) Both features contributes



(b) One feature is irrelevant.

Figure 2.10.1: Illustration of how the angle between the decision boundary and the feature axis can be used to determine the importance of a feature. In a, both angles are approximately the same. If one feature was removed the classification performance would decline. In b on the other hand, the decision boundary is almost parallel to feature 1. This indicates that feature 1 is more important than feature 2. Indeed, feature 2 could be removed and the 0-dimensional boundary would still be able to correctly separate the points.

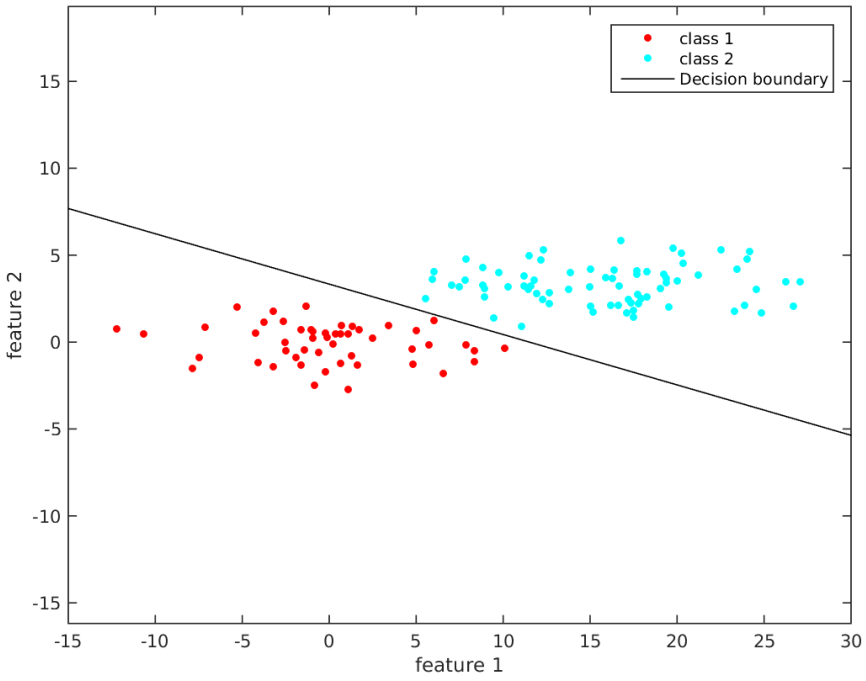


Figure 2.10.2: Illustration of scale bias in feature selection. The data is the same as in figure 2.10.1a, except that the numerical range of feature 1 has been increased by a factor 5. This makes feature 1 look more important than feature 2, even though the ability to discriminate is unaffected.

2.11 The Förstner interest point operator

Förstner (1986); Förstner and Gülch (1987) defines an *interest point operator* based on image gradients. Originally, the algorithm was developed for image correspondence problems. Here it is interesting for two reasons. Firstly, the interest point can be used to find a centroid for a region within an image, something that is required to compute some of the features used in this study. The second application is that the error estimate provided by the method in itself may be utilized as a measure of how radial, or cone shaped the tree-crown is.

The original work Förstner (1986); Förstner and Gülch (1987) introduces two related interest point operators, which will be called the *tangential interest point operator* and the *normal interest point operator*. Both operators derive the interest point from the image gradients as a least-square approximation of an *overdetermined linear system* composed of the gradients and positions of the gradients. However, the details when formulating the linear systems differs between the operators. This work will use the Roberts gradient defined in equation (2.3.10).

The following steps formulates the *normal equation* to solve for the *tangential interest point* $\boldsymbol{x}_0 = [x_0 \quad y_0]$. First, form the vectors of gradients $\boldsymbol{g}_x$ and $\boldsymbol{g}_y$, i.e. that contains the elements of x-wise gradient G_x and y-wise gradient G_y, respectively. Let

$$A = \begin{bmatrix} \boldsymbol{g}_x & \boldsymbol{g}_y \end{bmatrix}. \tag{2.11.1}$$

Let X and Y denote matrices of the x-wise and y-wise coordinates, respectively, of points in the image at which respective gradients are computed. Note that because the Roberts gradient is even in size, these coordinates correspond to points between pixels in the original image. For example, consider a 5×5 original image. Its corresponding Roberts gradients G_x, G_y are both 4×4 and

$$X = \begin{bmatrix} 1.5 & 1.5 & 1.5 & 1.5 \\ 2.5 & 2.5 & 2.5 & 2.5 \\ 3.5 & 3.5 & 3.5 & 3.5 \\ 4.5 & 4.5 & 4.5 & 4.5 \end{bmatrix}, Y = \begin{bmatrix} 1.5 & 2.5 & 3.5 & 4.5 \\ 1.5 & 2.5 & 3.5 & 4.5 \\ 1.5 & 2.5 & 3.5 & 4.5 \\ 1.5 & 2.5 & 3.5 & 4.5 \end{bmatrix}.$$

Next, let

$$\boldsymbol{b} = \boldsymbol{x} \hat{\cdot} \boldsymbol{g}_x + \boldsymbol{y} \hat{\cdot} \boldsymbol{g}_y, \tag{2.11.2}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ contains the coordinates of respective gradients, i.e. elements of $\boldsymbol{x}$ correspond to elements in X, in the same order as elements of $\boldsymbol{g}_x$ correspond to elements in G_x. The symbol $\hat{\cdot}$ denotes the element-wise product. With this setting, the tangential interest point is defined as the least square solution to the overdetermined linear system

$$A\boldsymbol{x}_0 = \boldsymbol{b}, \tag{2.11.3}$$

and is obtained from solving the systems corresponding normal equation for $\boldsymbol{x}_0$

$$A^T A \boldsymbol{x}_0 = A^T \boldsymbol{b}. \tag{2.11.4}$$

This concludes how the tangential interest point is computed. The difference in the derivation of the *normal interest point* is that the normal equation uses the setting

$$A = \begin{bmatrix} \boldsymbol{g}_x & -\boldsymbol{g}_y \end{bmatrix}, \tag{2.11.5}$$

and

$$\boldsymbol{b} = \boldsymbol{x}\hat{\cdot}\boldsymbol{g}_y - \boldsymbol{y}\hat{\cdot}\boldsymbol{g}_x, \tag{2.11.6}$$

instead of equations (2.11.1) and (2.11.2).

Given either the tangential- or the normal- interest point $\boldsymbol{x}_0$, the *least-square error* can be expressed with the following values. Firstly, the *residual* is defined as

$$\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}_0. \tag{2.11.7}$$

An estimate of the standard deviation corresponding to the least-square error is obtained from the residual as

$$\sigma_0 = \sqrt{\frac{\boldsymbol{r}^T\boldsymbol{r}}{n-2}}, \tag{2.11.8}$$

where n length of r. For a tree-crown, the σ_0 that correspond to to the normal interest point is a measure of the tree-crown radial pattern, and the σ_0 that correspond to the tangential interest point is a measure of tree-crown cone shape.

3 Materials

The dataset utilized in this study comes from ground measurements and aerial surveying of the Remningstorp test park in southern Sweden. Remningstorp is situated at $13.626°$ longitude and $58.466°$ latitude (WGS84), and lies 120 m above sea level.

The *ground measurements* were collected in 2014, by personnel at SLU. The measurements were collected at 261 stands, distributed over a square-grid pattern covering an area of approximately 10 km^2. The relative spacing was about 200 m in both the E-W and N-S directions. About each stand-point, measurements where taken on trees within a 10 m radius. Tree positions where measured by using a Trimble GeoExplorer 6000 GeoXR GPS-system, with an accuracy of below 1 m. Of the measured trees, species was noted for all trees, the diameter was measured for 95% of the trees, and height was measured for 28% of the trees. In total, measurements where taken on 6493 trees.

The *aerial survey* was conducted in the 14'th September 2014, by Blom Geomatics. The surveyed area covered approximately 47 km^2, and completely overlapped the area of ground measurements. The altitude of flight was 430 m above sea level. The plane was equipped with both a Leica RCD30-camera and a Riegl LMS-Q680i ALS-system.

The camera was equipped with a CH62 camera head and a NAG-D 3.5/50 lens with a focal length of 53 mm. 8-bit imagery was collected in the RGB bands and co-registered NIR (wavelength 780 to 880 nm) band. The size of an image was 9000×6732 pixels. At the ground level, each image covered about 305×230 m^2. The ground sampling distance is 3.40 cm. The images overlap with 60% in the direction of flight and 30% sideways.

A total of 7218 images were collected. See figure 3.0.1 for an example. Of all these images, only those that were the *nearest neighbour* of some stand was utilized, i.e. in terms if horizontal distance from the camera-centre to the stand-point. Furthermore, images where the nearest stand are farther away than 50 m were discarded. After these steps, 105 images remained that depicted at least one stand.

The ALS-System had a field of view of $45/60$ degrees and emitted rays of wavelength 1550 nm. The system uses a *consecutive line* scan pattern, and it operates at a scan frequency of up to 266 kHz. The accuracy of the scanning is expected to be below 0.3 meters in all directions. The resulting scan was a dense point cloud in world-coordinates.

Segmentation was conducted on the ALS data for a 15 cm radius around each stand-point. The segmentation scheme is described in (Holmgren and Lindberg, 2013). The result was produced and delivered by Johan Holmgren in person. A total of 12440 tree-crowns where segmented. The segments was given in world coordinates.

In order to match the imagery with the segments, relevant images were geo-referenced, using Terrasolid TerraPhoto. For some images the error in the geo-referencing was large enough for trees to end up in-between segments, e.g. see figure 3.1.3. The extent of these errors is unknown.

The resulting data; imagery, segments, and ground measurement for a single stand is illustrated in figure 3.0.2.

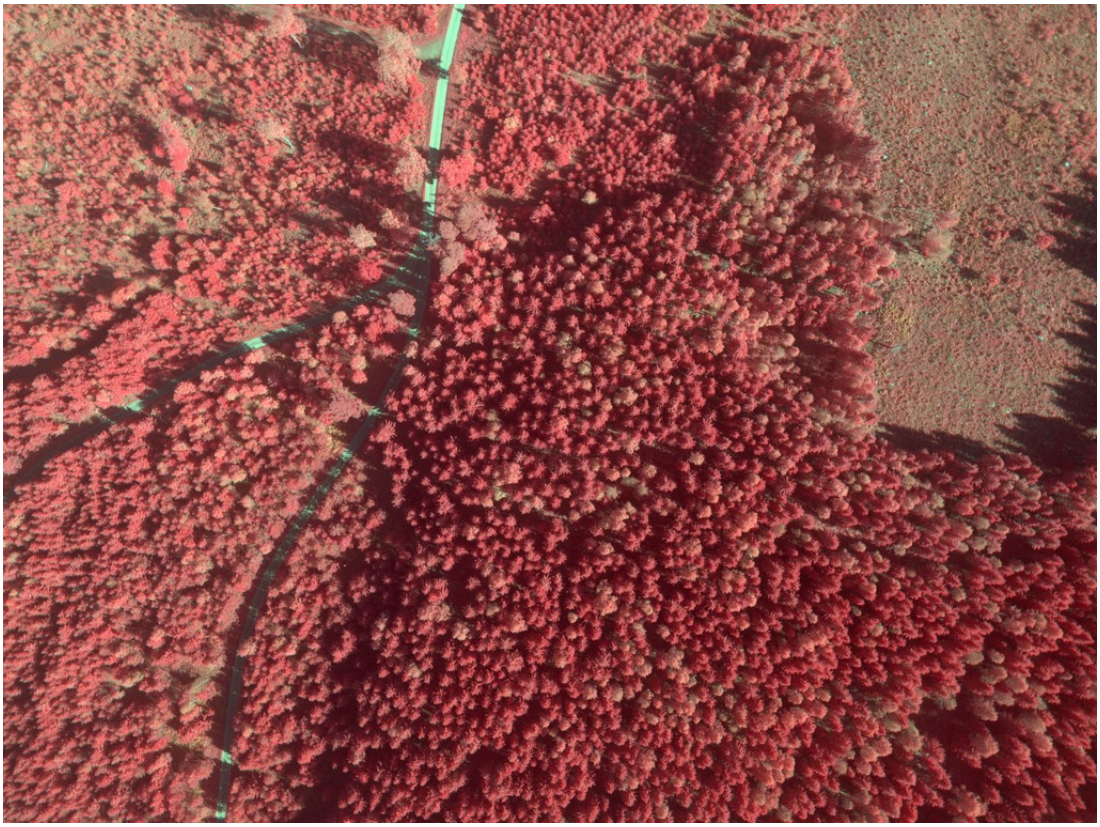


Figure 3.0.1: A sample CIR image from the image dataset. The original size was 9000×6732 *pixels (before geo-referencing). The NIR, red, and green channels are utilized.*

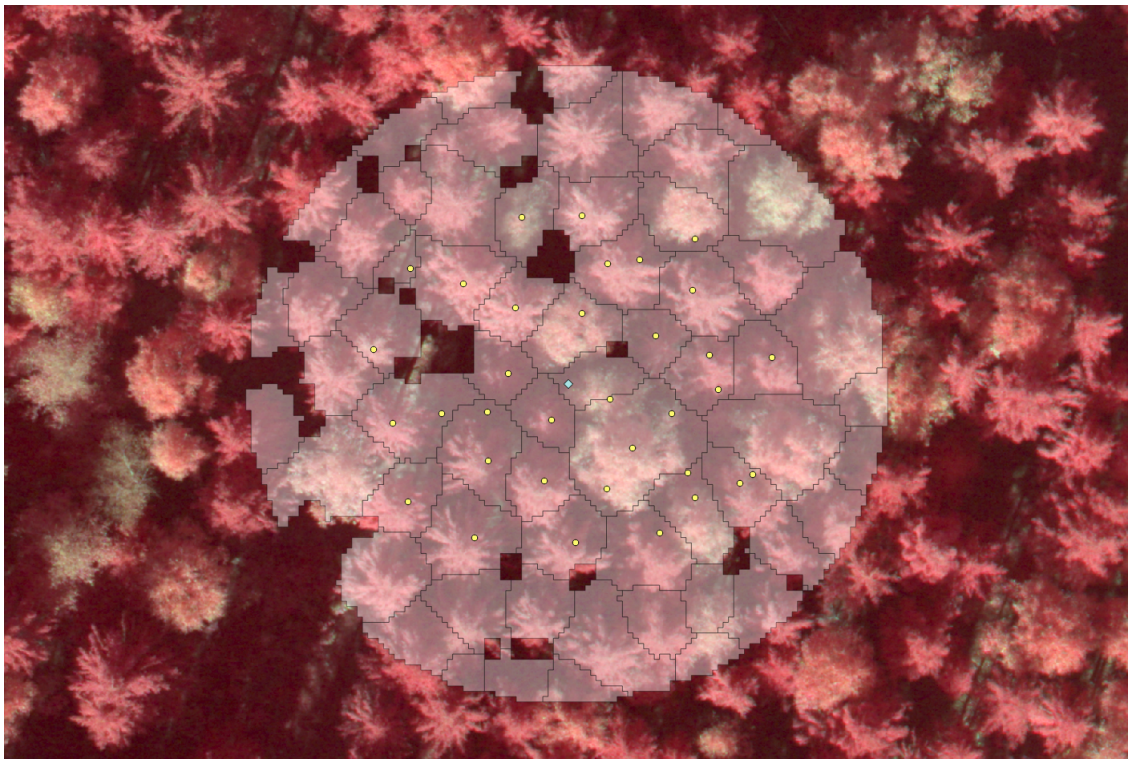


Figure 3.0.2: Imagery of a sample stand, with segments highlighted, and ground-measured trees marked as yellow dots. The cyan diamond (near the centre) marks the stand-point. In the raw-data there is a one-to-many mapping from tree-crown segments to ground-measured trees. As seen in the image, some segments are empty, while others contain multiple ground-measured trees. Also, a small offset introduced by the geo-referencing and the viewing geometry is seen. However, in this particular case it is small and hence not expected to have a significant impact on classification performance.

3.1 Extraction of single-tree level data

The goal of single-tree level data extraction is to represent the data as a set of trees, where each tree-crown is assigned a *segment*, a small image depicting the region covering the segment, and a single ground-truth species (either spruce, pine, or deciduous).

Firstly, all tree-crown segments that did not overlap a ground measured tree are removed along with segments that belong to a stand that was too far away from any camera-centre. Secondly, the remaining tree-crown segments are guaranteed to be inside an image and to overlap at least one tree. Thirdly, each tree-crown segment must be assigned a unique species. Any tree-crown segment that overlaps exactly one ground-measured tree is assigned the species of that tree. However, if a tree-crown segment overlaps multiple ground-measured trees, the species is assigned by the following procedure:

1. If any tree within the crown is has a height defined, pick the species of the tallest tree (with height defined).

2. Otherwise if a diameter is defined for any tree within the crown, pick the species of the widest tree (that has diameter defined).

3. In case the former steps did not resolve the tree-crown species, discard the crown.

For remaining tree-crowns, a single-tree-image is cut from the CIR channels of the aerial image containing the segment. The single-tree-image contains the imagery within the bounding box of the segment, plus a two-pixel-wide border.

In total, 1972 tree-crowns remained after this procedure. Each of these tree-crowns are assigned a CIR image, a mask corresponding to the segment, and a ground-truth species. Since the fraction of non-birch deciduous trees was too small for analysis, 72 such trees were removed. Thus the set of deciduous trees contains only birches. The data that was used in the analysis of the remaining 1900 tree-crowns is illustrated in the figures 3.1.1 and 3.1.2, for some sample single-trees. The species distribution is presented in figure 3.1.4. The size of the single-tree-images ranged from 63×47 pixels to 330×322 pixels (see figure 3.1.5).

3.2 Software

The platform used was 64-bit Ubuntu 14.04 LTS. The solution was implemented in MatLab R2015a. The SVM implementation from MatLabs *Statistics and Machine Learning Toolbox* was used with default parameters, except for the kernel function that was set to *linear*.

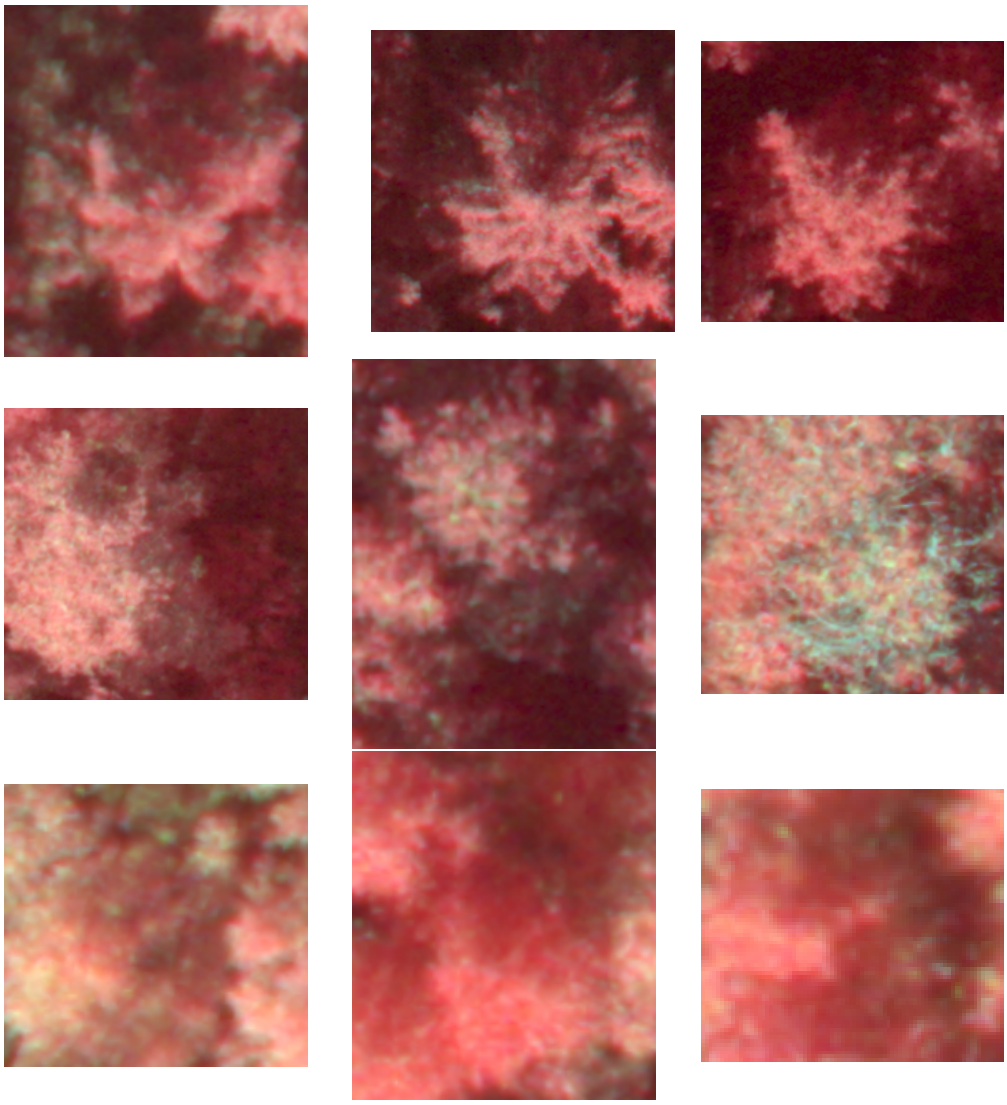


Figure 3.1.1: Sample extracted single tree images. The first row consists of spruces, the second row consists of pines, and the third row consists of deciduous trees.

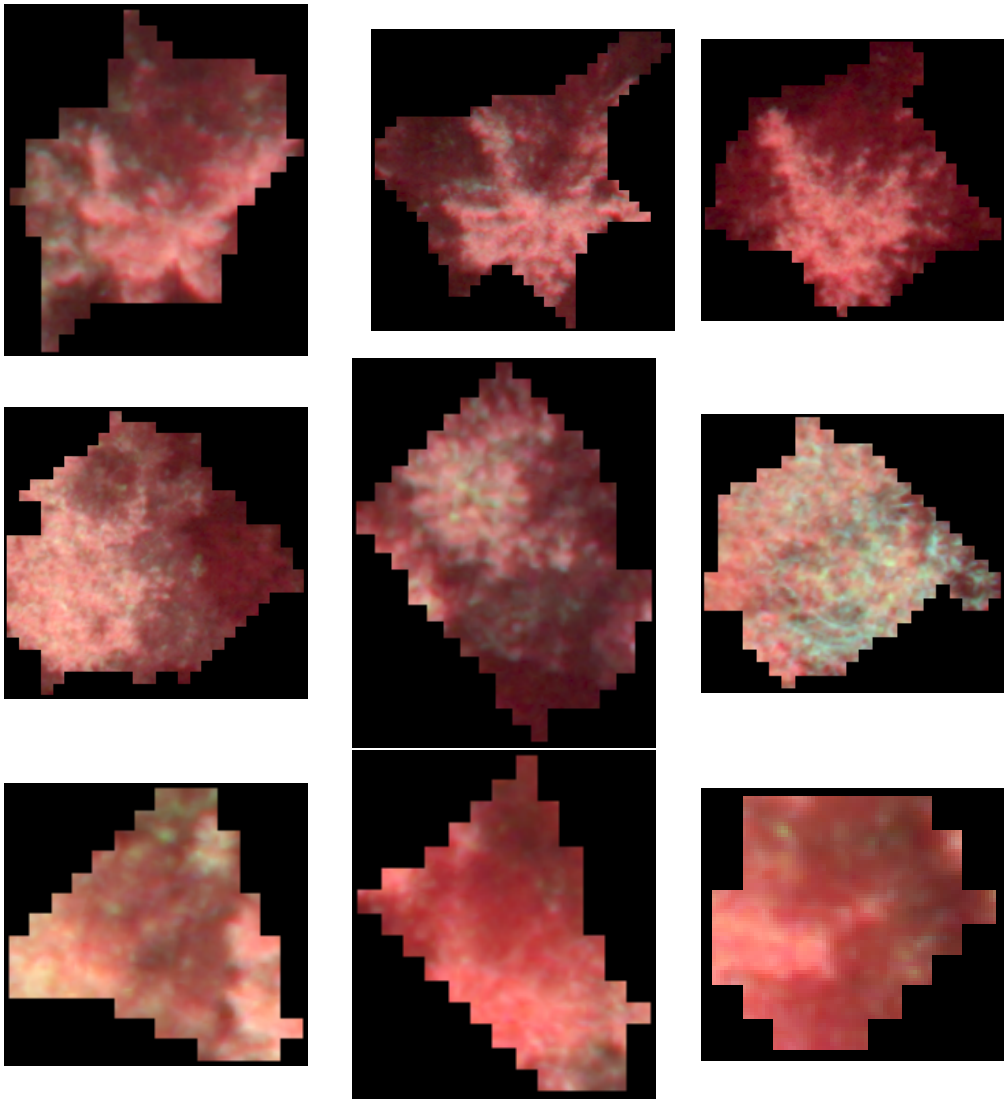


Figure 3.1.2: Same trees as in figure 3.1.1, but with masks applied. Features are derived from the masked (non-blackened) pixels.

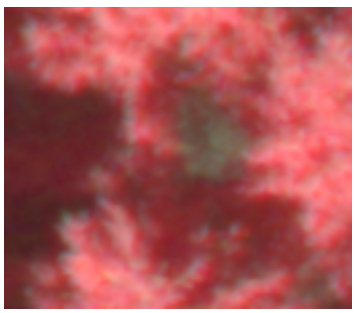
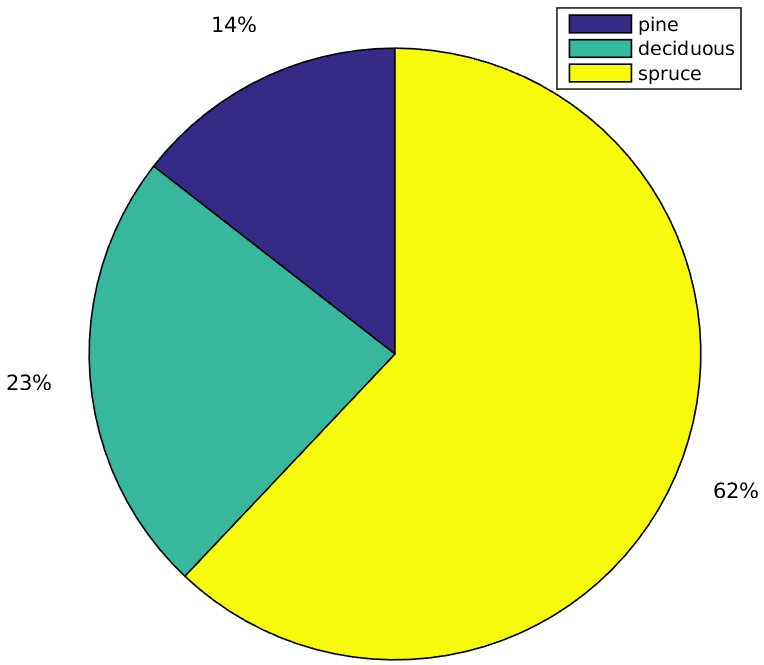


Figure 3.1.3: In some cases, the error in the geo-referenced image was so large that trees ended up in-between segments.



# pines	275
# deciduous	446
# spruces	1179

Figure 3.1.4: Distribution of species in the filtered data. Pine is the class that limits the results of the study.

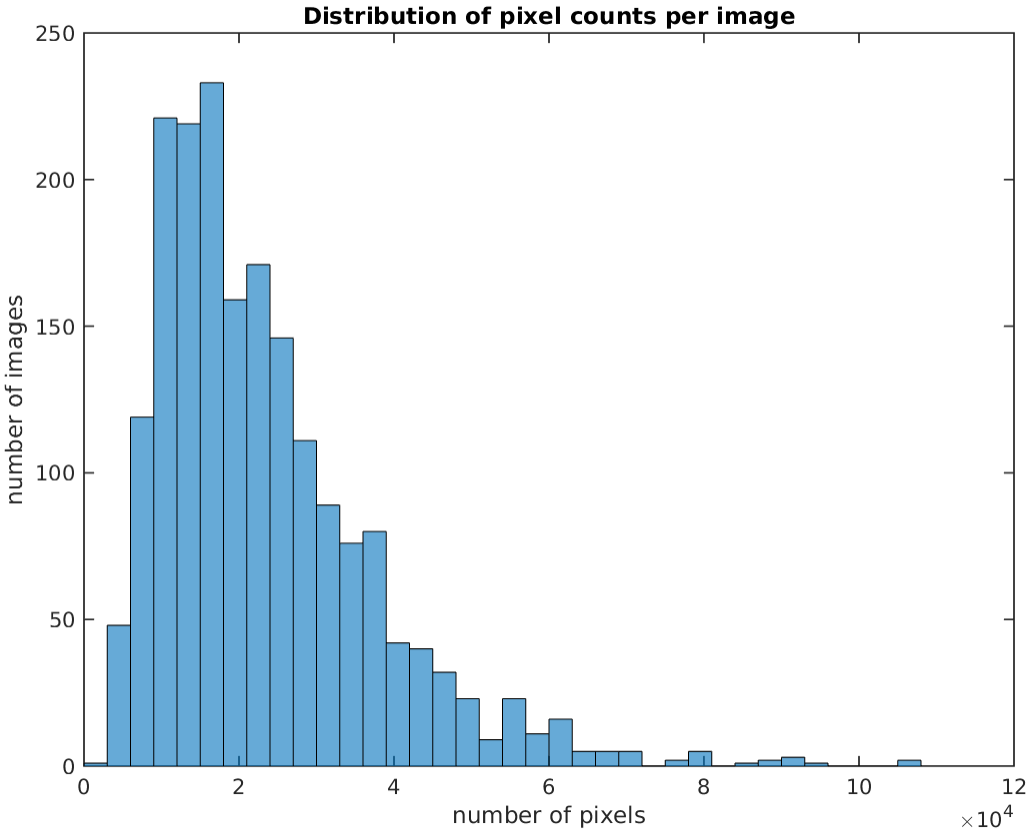


Figure 3.1.5: Distribution of image sizes (number of pixels). The largest single-tree-image is 330×322 *pixels, and the smallest image is* 63×47 *pixels. The average image is* 23415 *pixels in total (rounded), which roughly correspond to* 153×153 *pixels.*

4 Methods of feature extraction

Spatial features utilized in this study have been divided into the three categories: *second-order texture*, *morphological features*, and *gradient features* (derived from the image gradient). Some of these features are based on earlier work in *single-tree remote sensing* (STRS), but most of them are more general descriptors in the sense that they can be used to describe more general objects than just tree-crowns. Also, a mix of old and new (to STRS) applications of spectral features have been included for reference. Both spatial- and spectral- features are computed for the masked pixels in each colour channel (near infra-red (NIR), red, and green), separately. The features are summarized in table 4.0.1. The total number of distinct features per channel is 122, and the total number of features is 366.

4.1 Spectral features

Intensity mean, *lit-intensity mean*, and *top intensity* are from (Gougeon, 1995). A minor difference from the original work is that pixels used by *lit-intensity mean* is determined by the same channel that the feature is computed for. In the original work, the NIR channel determined which pixels to use. The features *standard deviation* and *entropy* have also been utilized in earlier work (Meyera et al., 1996; Leckie et al., 2003b; Brandtberg, 2002). For all spectral measures, all intensity levels were used, i.e. the histograms have 256 bins.

Table 4.0.1: Complete list of all features that where included in this study. The counts $\#/chan$ is the number of variations of the basic feature (per channel). For example, the variants may correspond to multiple measures taken from a histogram that is specific for the feature. Another example is when multiple variants correspond to different parameters to the features, such as varying step length. The bold-font figures of each respective set is the number of features within the set. The set of second-order textural features is predominantly larger than the others, because parameters could be varied to a large extent. The spectral features are described in section 4.1. The morphology features are defined in section 4.2. The gradient features are defined in section 4.3. The second-order textural features are defined in section 4.4.

spectral

name	**#/chan**	**description**
intensity mean	1	The mean intensity.
lit-intensity mean	1	The mean intensity of pixels with an intensity greater than the mean intensity of all pixels.
top intensity	1	The maximum pixel intensity.
intensity std	1	Estimated intensity standard deviation.
intensity entropy	1	Entropy of pixel intensities. See equation (2.5.5).
intensity skewness	1	Skewness, or standardized third moment of intensity distribution. See equation (2.5.4).
intensity kurtosis	1	Kurtosis, or standardized fourth moment of intensity distribution. See equation (2.5.4).
	7	

morphology

name	**#/chan**	**description**
branchstarness	4	Measures the radialness of the branch-pattern. The branch pattern is extracted from the single-tree imagery using morphological techniques. Two different centroids are tested.
granulometry	2	A (grey-level) morphological measure of the grain size distribution. It aims to capture sizes of leaf or twig bundles in the imagery.
	6	

gradient

name	**#/chan**	**description**
Förstner	2	Error measure for the for the Förstner interest point. See equation (2.11.8).
gradstarness	2	A measure of gradient diversion relative to a centroid. Similar to branchstarness.
gradanisotropy	2	Measures the amount of intensity change in principal directions.
gradentropy	1	The entropy of a gradient-based directional histogram.
	7	

2'nd-order texture

name	**#/chan**	**description**
GLCM	90	Measure derived from the GLCM. All measures in table 2.6.1 are used. Additionally three principal directions are tested and five different step-lengths.
Haar wavelets	12	A measure that aims to capture the amount of texture at different scales. The feature is computed for the three sets of difference-coefficients at three different scales. See equation (4.4.1).
	102	

4.2 Morphology features

Branchstarness is based on the measure of the radial branch pattern proposed by Brandtberg (1997, 2002). Figure 4.2.1 illustrates radialness on an intuitive level. The branchstarness feature is computed from a *branch pattern*, a binary image of individual branches (see figure 4.2.4). The branch pattern is extracted from a tree-crown image by applying this sequence of processing steps:

1. Apply an edge filter and threshold.
2. Skeletonize edges.
3. Prune the skeleton.
4. Remove branch points from the pruned skeleton.
5. Remove isolated pixels, with no 8-connected neighbours.

These steps are illustrated in the figures 4.2.3 and 4.2.4, the latter that shows the masked end result. In the first *edge filter* step the original tree-crown image is filtered by the Laplacian filter L_8 defined in equation (2.3.7). Of the filtered result, positive pixels are kept. In summary,

$$X_{edges} = (X * L_8) > 0, \tag{4.2.1}$$

where the output edge-image X_{edges} is a boolean image such that a pixel is set to true if it belongs to an edge, and zero otherwise. An example edge image is seen in figure 4.2.3b. The next step extracts the *morphological skeleton* (see figure 4.2.3c), that is just one pixel thick (Gonzalez, 2008, chapter 9.5.7, 11.1.7). Many algorithms for skeletonization exists which produce similar results. In this work, skeletons are generated by MatLabs function *bwmorph()*. The next step applies *morphological pruning* to prune off small branches known as *parasitic components* from the skeleton (Gonzalez, 2008, chapter 9.5.8). The length of parasitic components to remove is set to two pixels. The fourth step removes branch points from the pruned skeleton (once again using *bwmorph()*). After this step, the image will consist of isolated one-pixel-thick blobs called *branches* (see figure 4.2.3e). The fifth and final step removes isolated pixels.

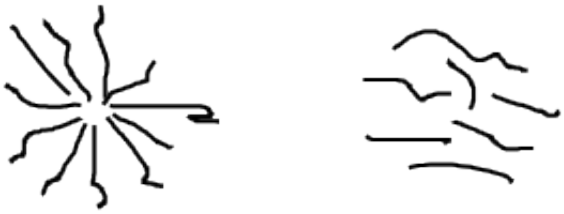
Once the branch pattern is extracted, each branch is analysed independently of the others. For each branch, the centre of mass is computed. Also the major- and minor-axis and respective major- and minor radii are all computed by *principal component analysis* on the n-normalized covariance matrix of pixel positions that belong to the branch. For each branch, the *angular diversion* relative to the tree stem is computed. This requires a centroid for representing the stem, or centre of the branch pattern. Formally, angular diversion is defined as the *smallest angle* between the line passing through the centroid and the branch centre-off-mass and the other line that that is major axis of the branch. This is illustrated in figure 4.2.2. Note that the angular diversions will be in the interval $[0°, 90°]$. As a final filtering step, branches that display insufficient directionality are removed. This is achieved by removing branches where the fraction of the major and minor radii is less than 1.5. Branchstarness is evaluated for two different centroids. The

first version uses the centre of mass of the masked branch pattern as seen in figure 4.2.4. The second version uses the tangential Förstner interest point defined in section 2.11.

After each branch has been processed the angular diversions are accumulated in a histogram, weighted by the major radius of the corresponding branch. The histogram h has 9 bins, each corresponding to 10° intervals. Finally, the histogram is normalized so that the sum of its elements equals 1. Two features are extracted from this histogram. The first is the *raw moment*, defined as

$$M(h) = \sum_{x=1}^{n} xh(x). \tag{4.2.2}$$

The second measure is the entropy of the histogram, defined in equation (2.5.5).



(a) Radial *(b) Not radial*

Figure 4.2.1: Illustration of radialness, which branchstarness aims to capture.

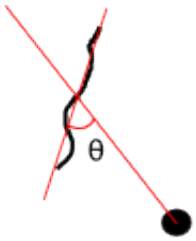
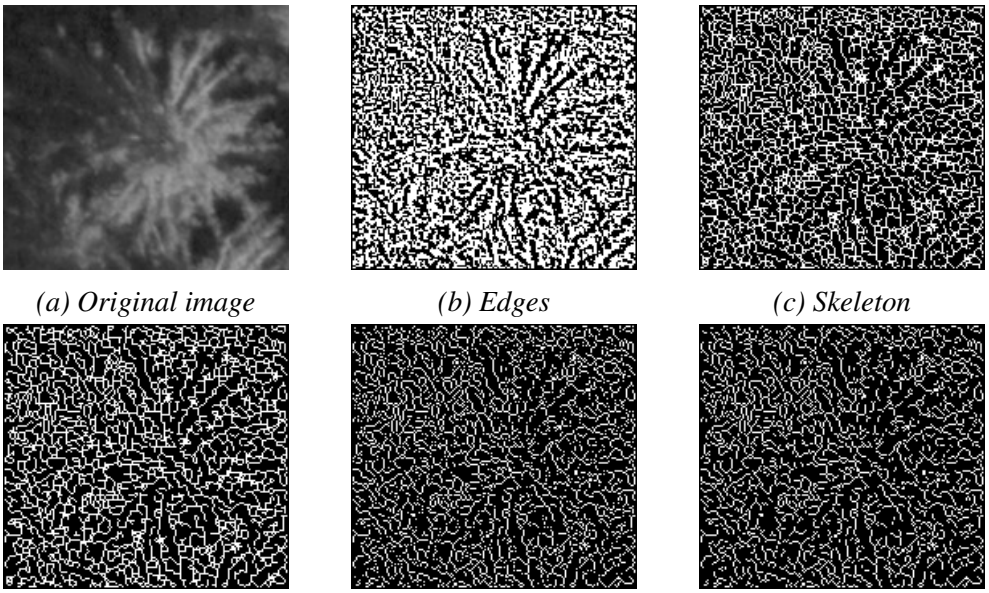


Figure 4.2.2: Illustration of angular diversion θ. The big dot is the centroid, the red lines are the line through the centroid, and the centre of mass of the branch; and the major axis of the branch, respectively. The smaller angle between the two lines is the angular diversion θ.



(a) Original image *(b) Edges* *(c) Skeleton*

(d) Pruned skeleton *(e) Removed branch points* *(f) Removed isolated pixels*

Figure 4.2.3: Steps for extracting the branch pattern.

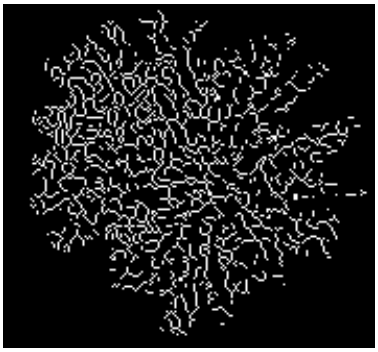


Figure 4.2.4: Masked result of branch extraction algorithm. Branchstarness is derived from this pattern and from a centroid.

(a) Low moment *(b) High moment* *(c) Entropy*

Figure 4.2.5: Illustration of what granulometry captures. The first figure has a low granulometry moment compared to the others because the grains are small. The second figure has a higher moment because the grains are larger. Both first figures have a low entropy because all grains are of the same size. The third figure has grains of different sizes and therefore is of higher entropy than the other examples.

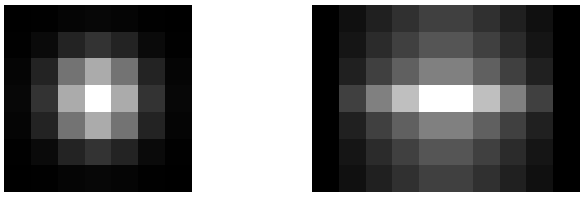
The granulometry feature aims to capture the grain size distribution in the tree crown (Gonzalez, 2008, p. 674). The concept of granulometry is illustrated in figure 4.2.5. A granulometric histogram is constructed by accumulating the total image intensity after applying *morphological erosion* with *disk shaped* structuring elements of various sizes. For each histogram bin, a particular structuring element radius is used, ranging from 1 to 35. To avoid ambiguity, the size of structuring-element-matrix corresponding to the disk will have the size $2r + 1 \times 2r + 1$, so that its centre always correspond to the centre of a pixel. The histogram is then normalized. The granulometry features consist of the raw moment and entropy of the granulometric histogram.

4.3 Gradient features

Gradient features are derived from the image gradient. This set of features therefore encompasses the *Förstner features* introduced in section 2.11. Specifically, the two Förstner features are defined as the σ_0's of equation (2.11.8), corresponding to the *tangential interest point* and the *normal interest point*. All the remaining features introduced in this section utilizes the *Sobel gradient*, defined in equation (2.3.12) (Note that Förstner centroids are still computed using the *Roberts gradient*). The respective gradients are denoted G_x and G_y (see equation 2.3.8).

Gradstarness is similar to Branchstarness that was introduced in the previous section. The difference is that in Gradstarness, the gradient replaces the morphological branch pattern. In Gradstarness, the angular diversion at a specific pixel at $[x \quad y]^T$ is the diversion of the gradient direction d_{ij} (see equation 2.3.13) and the vector from the centroid to the pixel $[x \quad y]^T - [x_0 \quad y_0]^T$. When accumulated in the histogram, the diversions are weighted by their corresponding magnitudes m_{ij}'s (see equation 2.3.14). Consecutive steps are similar to Branchstarness. The histogram raw moment and histogram entropy constitute the specific Gradstarness features, and Gradstarness uses the Förstner tangential centroid.

Gradanisotropy aims to capture the difference in the *amount of intensity change* between the direction that has the most- and least- intensity change, respectively (See figure 4.3.1 for an example). It is computed by *principal component analysis* on the gradients. Initially, the covariance matrix centred at $(0, 0)$ (i.e. the mean is set to be zero in the x and y direction) is computed for the gradient values. From the covariance matrix the eigenvalues λ_1, λ_2 such that $\lambda_1 \geq \lambda_2$, are obtained from the singular value decomposition (svd). The *anisotropy measure* is then defined as λ_2/λ_1. Note that this means that smaller values of the feature correspond to greater anisotropy. Another measure that is also included in Gradanisotropy is the *non-systematic anisotropy measure*. This measure attempts to remove systematic intensity change in the image (i.e. one end of the image being lighter than the other). This is achieved by subtracting the x-wise averages from G_x and the y-wise averages from G_y before forming the covariance matrix.



(a) Isotropic *(b) Anisotropic*

Figure 4.3.1: Illustration of what the anisotropy feature aims to capture. In a, the change in intensity is similar over all principal directions of the image. In b, the change is larger column-wise, than it is row-wise. The anisotropy feature of the first image is 1*, indicating that it is isotropic. The feature of the second image is* 0.398*, which indicates anisotropy.*

The Gradentropy feature is computed from a bidirectional histogram of gradient directions. Bi-directionality is achieved by reversing every gradient with an angle outside

the interval $[-\pi, \pi)$, relative to the x-axis. After the reversal step, the gradients are accumulated into a directional histogram, weighted by their magnitude. The histogram uses twelve bins, each corresponding to $15°$. From this histogram, the entropy is extracted.

4.4 Second-order texture

Two approaches to second-order textural features are tested. Features derived from the GLCM and features derived from Haar wavelet coefficients.

For a normalized and bi-directional GLCM (see section 2.6), all six features defined in table 2.6.1 are included. Additionally, these features are computed for the step-lengths $S = \{1, 2, 4, 8, 16\}$, in three principal directions: horizontally, vertically, and diagonally. That is the offsets are $O = \{(0, s), (s, 0), (s, s)\}$, for all $s \in S$.

The measure utilized for wavelets aims to capture the *amount of texture* in principal directions. For an arbitrary level, let A denote the matrix of approximation coefficients and D any of the three sets of difference coefficients (horizontal, vertical, or diagonal). The feature is defined as

$$f(x) = \sqrt{\sum_{i,j} \frac{d_{i,j}^2}{a_{i,j}^2}}. \tag{4.4.1}$$

A minor difference with the introduction to wavelets provided in section 2.7, is that the non-orthogonal wavelet basis $\{\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}1\\-1\end{bmatrix}\}$ is used instead of $\{\frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix}, \{\frac{1}{\sqrt{2}}\begin{bmatrix}1\\-1\end{bmatrix}\}$. The feature is computed for the first four levels of decomposition, and for all three sets of difference coefficients.

5 Experiments

The objective of this study is to *evaluate a large set of spatial features* for the three species tree-crown classification problem, and to investigate what spectral information is required for classification with spatial features. Both of these issues are investigated in experiment 1 that uses feature ranking to establish the relative importance of each feature. The redundancy of each feature is also investigated by successively removing the worst-feature (based on the ranking) and estimating the loss of the n'-best features subset. To establish if spectral features are necessary, the experiment is re-run on different subsets of features. In particular, the loss from removing spectral features and colour channels is measured by this scheme.

In experiment 1, the best-features are selected from all data, including the test data used to estimate classification performance. This approach is sound in the sense that the goal is to find the features that best describe the whole dataset, but introduces bias in the sense that the performance results will not correspond to a real situation. The remedy is to rank features on training data exclusively. The purpose of experiment 2 is to estimate the *unbiased classification performance* that is expected in practice, using the best features. The unbiased result is a complement to experiment 1.

Experiment 3 asserts the feasibility of the voting procedure. Given this experiments result and the result from experiment 2, it can be assessed if the classification method chosen for this study was feasible.

All the following experiments uses algorithm (3) with the normalization scheme defined at the end of section 2.10.1, to rank all features for all three classification pairs.

5.1 Classification with multiple classes

The classification task involves the three classes spruce, pine, and deciduous trees. Because the SVM is a binary classifier it must be extended handle more than two classes. This study takes the one-versus-one approach, because it is simple, and also because the result of one-versus-one classification is more informative than the result of one-versus-all classification. That is, the one-versus-all method would not be able to detect if the features that discriminates class A from class B are different from the ones that discriminate class A from class C. In order to aggregate the result from the three partial-SVMs, the voting scheme defined in section 2.9.1 is used with the addition that for unresolved cases (where each tree class obtains one vote) the class is instead determined by the partial-SVM that maximizes the absolute value of the *decision function*.

Even though the entity consisting of the partial-SVMs and the voting scheme will be referred to as a multi-SVM, the features used will be different for different partial-SVMs.

5.2 Normalized cross-validation

In the experiments that follows, classification performance is evaluated using a normalized version of cross validation described in section 2.8.3. The difference is that in each fold, the training data X_{train} is normalized using equation 2.10.1. The test data X_{test} is also normalized, but with the means and standard deviations from the columns of X_{train}. The reason for conducting this normalization is to account for the normalization that is applied when ranking the features (see section 2.10.1).

In all experiments, 10-fold cross-validation is used.

5.3 Experiment 1: Identifying the best features

Given the $m \times n$ *feature matrix* X (rows correspond to distinct observations and columns correspond to distinct features) and the *ground truth* vector of m class labels $\boldsymbol{y}$. For number of best-features n' (determined by ranking) from 1 to n, use *normalized cross-validation* on a multi-SVM (see sections 5.1, and 5.2) to evaluate the classification performance of each n'-best features subset. The class-normalized loss (equation 2.8.9) is utilized as the performance metric. It is computed from the compound predictions produced by cross-validation, i.e. not for each fold (see section 2.8.3 for explanation).

An important detail is that the same cross-validation partitions is used in the evaluation of each n'-best-features subset. The reason for this is to remove noise caused by the differences in the partitions.

5.3.1 Datasets

The experiment is re-run, first for two different sets of observations, then for three different sets of features. The purpose of the first two datasets is to obtain general results on which features are best, but also to investigate which subset of observations is the most suitable in successive runs. The first set of observations named ALL OBSERVATIONS incorporate all the 1900 tree-crowns available. The second set of observations named EQUAL FRACTIONS includes 825 randomly chosen, but distinct tree-crowns such that the number of observations of each class is the same, i.e. 275 tree-crowns of spruce, pine, and deciduous trees, respectively. The results are compared in order to determine which of the following factors that affect the classification performance the most. (1) To have unequal class fractions leading to bias for the most probable class, or (2) to base the results on less observations. Note that for EQUAL FRACTIONS, the class-normalized classification loss is equivalent to the simpler classification loss described in equation (2.8.7).

Once the set of observations yielding the best performance is determined, the algorithm is run for the following subset of features, in order to determine what spectral information is necessary:

- Spectral features.
- Spatial features.
- Spatial features in the red channel.

Table 5.3.1: Average amount of texture for all observations at different channels and scales. Values were computed using the wavelet feature defined in equation (4.4.1)*. At each level, the texture components corresponding to respective set of loss coefficients (i.e. horizontal, vertical, and diagonal) have been combined. The red channel displays the largest amount of texture on all levels of wavelet-decomposition.*

	level 1	**level 2**	**level 3**	**level 4**
NIR	0.0292	0.0541	0.0929	0.1412
Red	0.1613	0.2050	0.2553	0.3049
Green	0.0979	0.1519	0.2072	0.2566

The reason that the red channel was selected is that it maximized the amount of texture over all tree-crown observations (see table 5.3.1).

5.4 Experiment 2: Nested feature selection

For a fixed number of best features n', normalized cross-validation with a multi-SVM is used to produce the *confusion matrix*. In this experiment, the features are ranked on the *training data* inside each fold of cross-validation. For reference, the cross-validation is also run with feature selection from all data (like in experiment 1) for the same number of features n'. By taking the difference in the respective losses, the bias can be estimated.

5.5 Experiment 3: Feasibility of the voting procedure

For n' best features, normalized cross-validation is run as in experiment 1 (with feature ranking on all data), but instead of measuring performance the votes are counted for each class and for each observation. The votes are used to establish the fraction of *resolved elections* where the predicted class received the majority of the votes. The compound fraction of resolved predictions is the total fraction of resolved election. Partial fractions of resolved predictions are computed for each class as the fractions of resolved elections, were the corresponding observation belonged to the particular class in the ground truth. Note that the partial fractions does not take into account if the correct class was elected.

6 Results

6.1 Experiment 1

The results of running experiment 1 on the two different sets of observations ALL DATA and EQUAL FRACTIONS, is seen in the figures 6.1.1 and 6.1.2, respectively. Figure 6.1.2 reveals that the compound loss is lower for the EQUAL FRACTIONS set of observations, except at the left most part of the tail (number of best-features is $n' \geq 200$). Therefore, the remaining runs of this experiment and the successive experiments utilizes the EQUAL FRACTIONS set of observations.

The top-30 features for the respective classification pairs; spruce-versus-pine, spruce-versus-deciduous, and pine-versus-deciduous, are listed in the tables 6.1.1, 6.1.2, and 6.1.3, respectively. All of these lists are dominated by second-order textural features, in different principal directions and at different scales. The number of spectral features on the lists varies between two and six. Of the three classification pairs, the two involving deciduous trees has more spectral features on the list than spruce-versus-pine. Spectral features dominates the top-3 lists.

The compound performance of each feature subset listed in section 5.3.1 is seen in the figures 6.1.3 and 6.1.4. The figures reveal a penalty when the number of best-features is $n' < 35$, e.g. for $n' = 20$ the loss of the spatial feature set is 2.9 pp[18] higher than the corresponding loss for all features. For greater $n \geq 35$ the loss-plots of the two feature sets coincide. The first time they intersect is at $n' = 33$. The set of spatial features that are limited to a single channel stands out as having the highest loss. At $n' = 20$ the loss corresponding to that set was 13.0 pp higher than the loss corresponding to the set of all features.

The performance of the partial classification when using only spatial features is seen in 6.1.5, and the respective top-30 spatial features list is found in tables 6.1.4, 6.1.5, and 6.1.6. For spatial features only, a similar mix of wavelet and GLCM features dominates the top-30 lists as for using all features. A difference is that the Förstner features for spruce-versus-pine climbed to positions near five. Also, for pine-versus-deciduous classification, the Förstner features appears near the bottom of the list.

[18] Percentage points (pp) is the unit of arithmetic difference between percentages. For example, the difference between 1% and 5% is 4 pp.

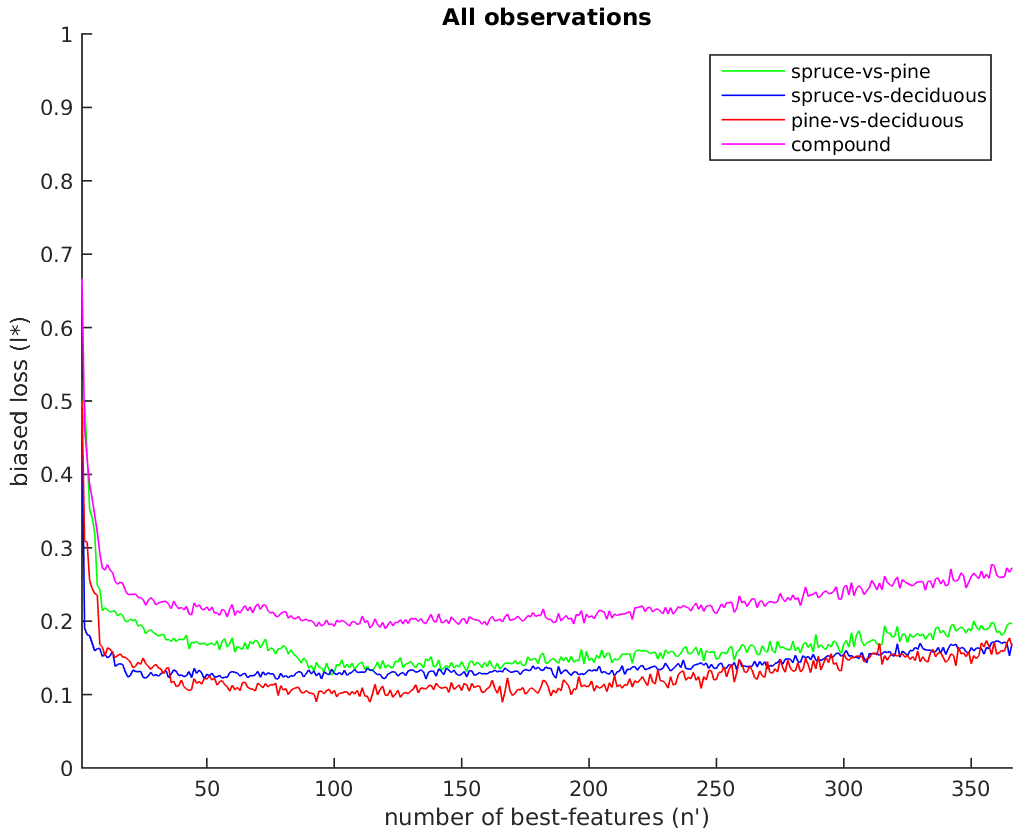


Figure 6.1.1: Result of experiment 1, using the ALL DATA *set of observations. The plot shows the biased loss for for all n'-best features subsets. If the plot is read from right to left, each decrement of n' correspond to removing the worst features determined by feature ranking. All plots within the figure displays J-curves. Removing features initially decreases the loss because the influence of overfitting lessens. In the flat sections, e.g. for $n' \in [100, 175]$, features are redundant, so that removing the worst features has little effect on the loss. However, when the number of features is low, $n' \leq 26$, a steep increase in loss is seen, because each feature adds significant information to the classifier. The height of respective plots illustrates the difficulty of classification, partials and compound. Spruce-versus-pine stands out as being the hardest pair to classify. The compound loss is higher than the partial losses, because in general, discriminating three classes is harder than discriminating two classes.*

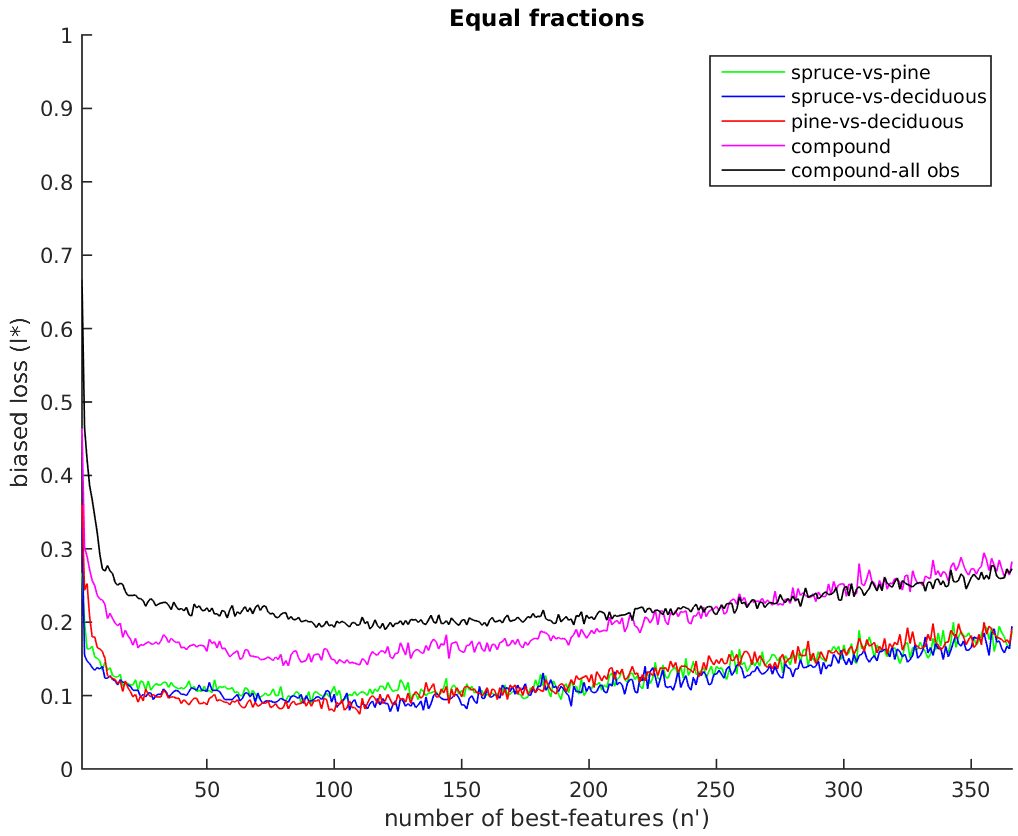


Figure 6.1.2: Result of experiment 1, using the EQUAL FRACTIONS *set of observations. Additionally, the compound result using* ALL DATA *is included for comparison (from figure 6.1.1). Clearly, the compound loss using* EQUAL FRACTIONS *is below that of using* ALL DATA *(except in the $n' \geq 200$ portion of the tail), as seen in the height of the plots. Also, for* EQUAL FRACTIONS *the losses of partial classification coincides. This suggests that all species are similarly difficult to discriminate.*

Table 6.1.1: A result from experiment 1. Best features for spruce-versus-pine classification using the EQUAL FRACTIONS *set of observations. k denotes the level of wavelet-decomposition and $\boldsymbol{q}$ denotes the GLCM-offsets. The table is dominated by second-order textural measures at different scales and in different principal directions. The only spatial feature on the list that is not a measure of texture is the Förstner feature. The two spectral features on the list are highlighted in grey.*

rank	channel	feature
1	Red	intensity mean
2	NIR	GLCM-contrast, $\boldsymbol{q} = [8, 0]$
3	NIR	wavelet-horizontal $k = 1$
4	NIR	GLCM-contrast, $\boldsymbol{q} = [0, 2]$
5	NIR	wavelet-vertical $k = 2$
6	NIR	skewness
7	Red	GLCM-energy $\boldsymbol{q} = [8, 0]$
8	Red	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
9	NIR	GLCM-correlation $\boldsymbol{q} = [1, 0]$
10	Green	GLCM-contrast $\boldsymbol{q} = [0, 1]$
11	Red	GLCM-contrast $q = [0, 4]$
12	Green	wavelet-vertical $k = 1$
13	Green	wavelet-diagonal $k = 1$
14	Red	GLCM-contrast $\boldsymbol{q} = [8, 8]$
15	NIR	GLCM-correlation $\boldsymbol{q} = [2, 0]$
16	Green	GLCM-correlation $\boldsymbol{q} = [8, 8]$
17	Green	GLCM-contrast $\boldsymbol{q} = [4, 0]$
18	NIR	GLCM-contrast $\boldsymbol{q} = [0, 1]$
19	NIR	GLCM-contrast $\boldsymbol{q} = [4, 0]$
20	NIR	wavelet-diagonal $k = 1$
21	NIR	GLCM-homogeneity $\boldsymbol{q} = [0, 16]$
22	Red	GLCM-contrast $\boldsymbol{q} = [0, 8]$
23	Red	GLCM-homogeneity $\boldsymbol{q} = 16$
24	NIR	wavelet-horizontal $k = 2$
25	Green	förstner-tangential
26	Red	förstner-normal
27	Green	wavelet-horizontal $k = 1$
28	Green	wavelet-horizontal $k = 4$
29	Red	wavelet-horizontal $k = 4$
30	Red	wavelet-vertical $k = 2$

Table 6.1.2: A result from experiment 1. Best features for spruce-versus-deciduous classification using the EQUAL FRACTIONS *set of observations. The mix of spatial features is similar to table 6.1.1, except that the Förstner features do not appear. In total, six spectral features appear on the list.*

rank	channel	feature
1	Red	lit-intensity mean
2	Green	lit-intensity mean
3	Red	intensity mean
4	NIR	GLCM-contrast $\boldsymbol{q} = [2, 0]$
5	Red	GLCM-contrast $\boldsymbol{q} = [1, 1]$
6	Red	GLCM-contrast $\boldsymbol{q} = [0, 4]$
7	Red	GLCM-contrast $\boldsymbol{q} = [0, 16]$
8	Red	GLCM-homogeneity $\boldsymbol{q} = [4, 4]$
9	Red	GLCM-homogeneity $\boldsymbol{q} = [1, 0]$
10	Green	GLCM-max(P) $\boldsymbol{q} = [16, 0]$
11	Green	GLCM-max(P) $\boldsymbol{q} = [1, 0]$
12	NIR	GLCM-max(P) $\boldsymbol{q} = [0, 2]$
13	NIR	GLCM-max(P) $\boldsymbol{q} = [0, 8]$
14	Green	intensity entropy
15	Green	intensity standard deviation
16	Green	GLCM-max(P) $\boldsymbol{q} = [1, 1]$
17	Red	intensity entropy
18	Green	GLCM-entropy $\boldsymbol{q} = [16, 16]$
19	Red	wavelet-diagonal $k = 4$
20	Green	wavelet-diagonal $k = 4$
21	NIR	GLCM-correlation $\boldsymbol{q} = [1, 0]$
22	Red	GLCM-correlation $\mathbf{1} = [4, 4]$
23	Green	GLCM-max(P) $\boldsymbol{q} = [4, 4]$
24	NIR	GLCM-contrast $\boldsymbol{q} = [4, 4]$
25	NIR	wavelet-diagonal $k = 3$
26	NIR	wavelet-horizontal $k = 3$
27	Green	GLCM-max(P) $\boldsymbol{q} = [0, 1]$
28	Green	GLCM-energy $\boldsymbol{q} = [16, 0]$
29	NIR	GLCM-energy $\boldsymbol{q} = [8, 0]$
30	NIR	GLCM-contrast $\boldsymbol{q} = [4, 0]$

Table 6.1.3: A result from experiment 1. Best features for pine-versus-deciduous classification using the EQUAL FRACTIONS *set of observations. The composition of features is similar to table 6.1.2.*

rank	channel	feature
1	NIR	GLCM-homogeneity $\boldsymbol{q} = [4, 0]$
2	NIR	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
3	Green	lit-intensity mean
4	Red	lit-intensity mean
5	NIR	wavelet-diagonal $k = 3$
6	NIR	GLCM-homogeneity $\boldsymbol{q} = [2, 0]$
7	NIR	GLCM-contrast $\boldsymbol{q} = [8, 0]$
8	Red	intensity mean
9	Green	GLCM-homogeneity $\boldsymbol{q} = [1, 0]$
10	NIR	intensity top
11	NIR	GLCM-correlation $\boldsymbol{q} = [2, 0]$
12	Green	intensity mean
13	Green	GLCM-correlation $\boldsymbol{q} = [0, 1]$
14	NIR	GLCM-homogeneity $\boldsymbol{q} = [2, 2]$
15	Green	GLCM-entropy $\boldsymbol{q} = [16, 0]$
16	NIR	GLCM-entropy $q = [0, 1]$
17	Green	GLCM-homogeneity $\boldsymbol{q} = [16, 16]$
18	Green	GLCM-homogeneity $\boldsymbol{q} = [4, 4]$
19	NIR	GLCM-correlation $\boldsymbol{q} = [4, 4]$
20	Red	GLCM-energy $\boldsymbol{q} = [16, 16]$
21	Red	GLCM-correlation $\boldsymbol{q} = [0, 8]$
22	Green	GLCM-contrast $\boldsymbol{q} = [16, 16]$
23	Green	GLCM-contrast $\boldsymbol{q} = [0, 16]$
24	Red	GLCM-contrast $\boldsymbol{q} = [0, 1]$
25	Green	GLCM-contrast $\boldsymbol{q} = [0, 4]$
26	Red	wavelet-diagonal $k = 4$
27	Green	wavelet-diagonal $k = 4$
28	NIR	GLCM-contrast $\boldsymbol{q} = [16, 16]$
29	Green	GLCM-homogeneity $\boldsymbol{q} = [16, 0]$
30	NIR	GLCM-contrast $\boldsymbol{q} = [1, 0]$

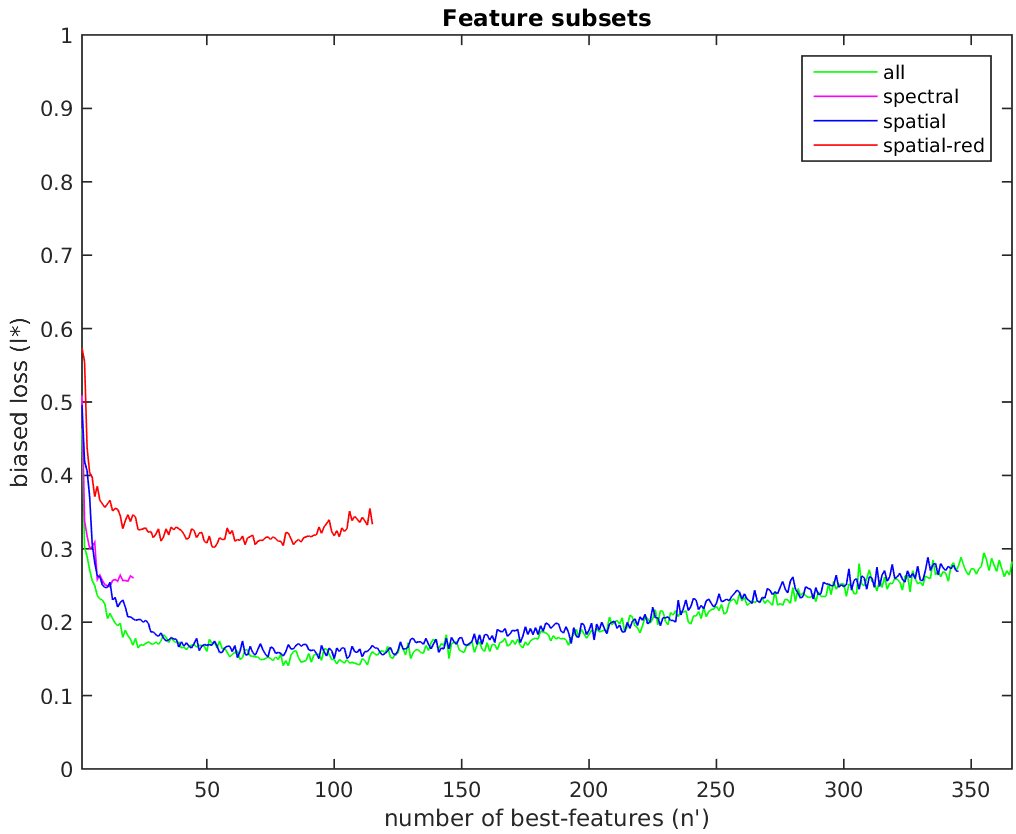


Figure 6.1.3: A result from experiment 1. Compound results for different subsets of features. The plot labelled 'all' refers to the compound loss seen in figure 6.1.2. The results are based on the EQUAL FRACTIONS *set of observations. The reason that the plots ends at different number of best-features n' is the different sizes of respective feature subsets. Spatial features limited to the red channel stands out as the worst set of features. If the number of best-features is sufficiently large, $n' \geq 35$, the loss when spatial features are used coincides with the loss when all features are used.*

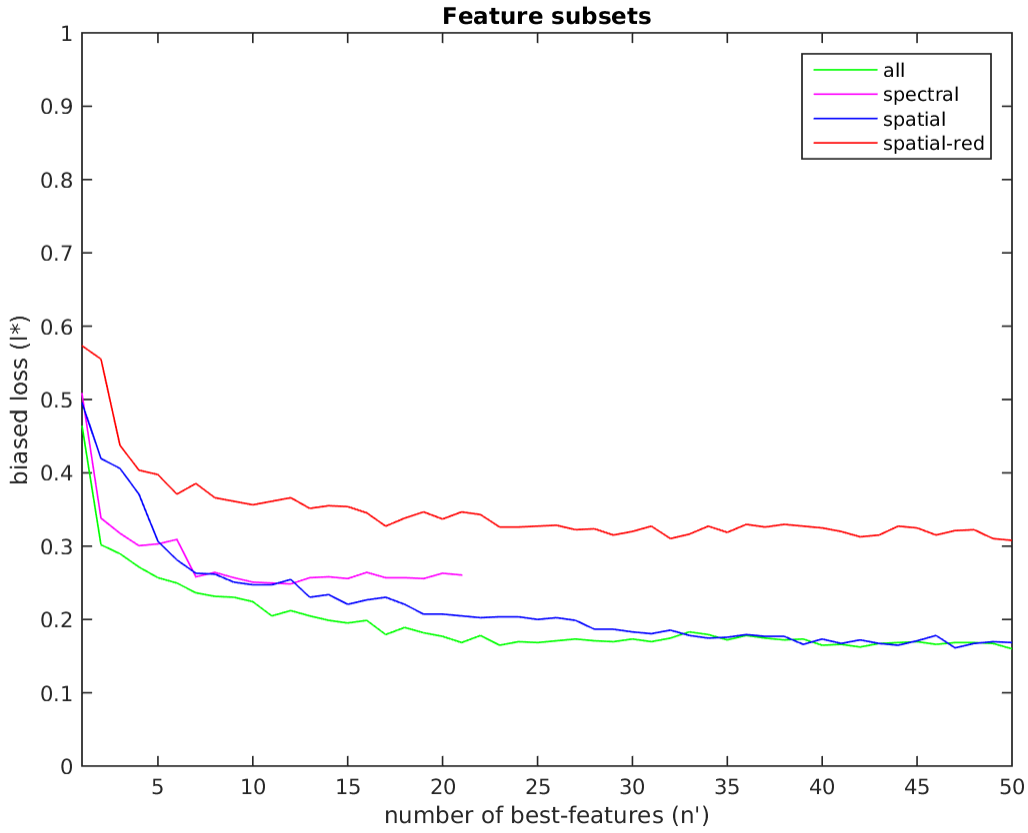


Figure 6.1.4: Same result as in figure 6.1.3, but zoomed in on on the part where at most $n' = 50$ features are used. This highlights the behaviour before the performance of the n'-best spatial features coincides with the performance of n'-best features (from all categories).

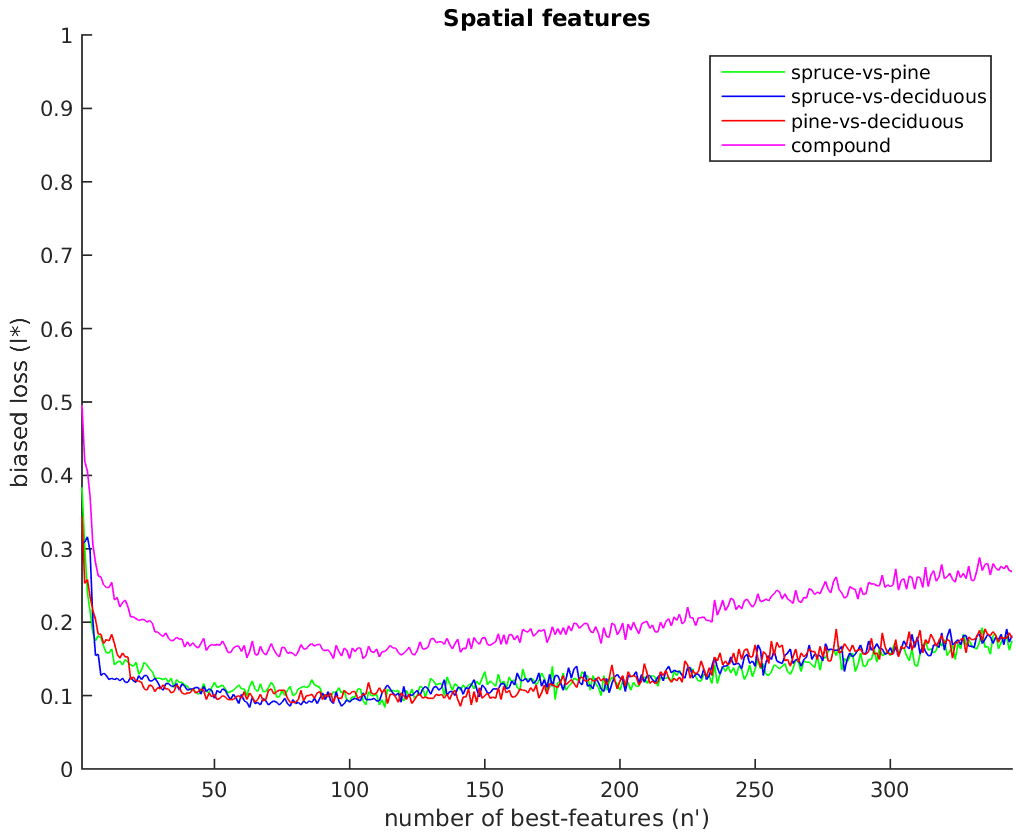


Figure 6.1.5: A result from experiment 1. Partial and compound performance when only spatial features are utilized. The result is based on the EQUAL FRACTIONS *set of observations. The performance result is similar to that in figure 6.1.2. However, a minor difference is that the partial results are more spread for small numbers of best-features ($n' \leq 20$).*

Table 6.1.4: A result from experiment 1. Best spatial features for spruce-versus-pine classification using the EQUAL FRACTIONS *set of observations. The mix of second-order texture is similar to when all features where included (see table 6.1.1). One difference is that the Förstner features have climbed in position from* 25 *to* 5.

rank	channel	feature
1	NIR	wavelet-vertical $k = 3$
2	NIR	GLCM-contrast $\boldsymbol{q} = [0, 2]$
3	NIR	GLCM-contrast $\boldsymbol{q} = [8, 0]$
4	NIR	wavelet-horizontal $k = 2$
5	Green	förstner-tangential
6	Red	förstner-normal
7	Red	GLCM-max(P) $\boldsymbol{q} = [4, 4]$
8	Green	GLCM-homogeneity $\boldsymbol{q} = [0, 2]$
9	Red	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
10	Red	GLCM-contrast $\boldsymbol{q} = [8, 8]$
11	Green	GLCM-contrast $\boldsymbol{q} = [0, 8]$
12	NIR	GLCM-correlation $\boldsymbol{q} = [2, 0]$
13	Green	GLCM-correlation $\boldsymbol{q} = [8, 8]$
14	NIR	GLCM-contrast $\boldsymbol{q} = [4, 0]$
15	Red	wavelet-vertical $k = 1$
16	Red	wavelet-diagonal $k = 2$
17	NIR	GLCM-homogeneity $\boldsymbol{q} = [2, 0]$
18	NIR	GLCM-max(P) $\boldsymbol{q} = [4, 0]$
19	Green	GLCM-correlation $\boldsymbol{q} = [0, 1]$
20	NIR	GLCM-max(P) $\boldsymbol{q} = [0, 16]$
21	NIR	GLCM-homogeneity $\boldsymbol{q} = [0, 16]$
22	Red	GLCM-correlation $\boldsymbol{q} = [0, 16]$
23	NIR	GLCM-correlation $\boldsymbol{q} = [4, 0]$
24	NIR	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
25	Green	GLCM-contrast $\boldsymbol{q} = [0, 2]$
26	Red	GLCM-contrast $\boldsymbol{q} = [0, 4]$
27	Green	GLCM-contrast $\boldsymbol{q} = [1, 0]$
28	NIR	wavelet-horizontal $k = 1$
29	NIR	wavelet-diagonal $k = 1$
30	Green	GLCM-contrast $\boldsymbol{q} = [0, 1]$

Table 6.1.5: A result from experiment 1. Best spatial features for spruce-versus-deciduous classification using the EQUAL FRACTIONS *set of observations. The mix of second-order texture is similar to when all features where included (see table 6.1.2).*

rank	channel	feature
1	Red	wavelet-vertical $k = 4$
2	Green	wavelet-vertical $k = 4$
3	Red	GLCM-correlation $\boldsymbol{q} = [4, 4]$
4	Green	GLCM-correlation $\boldsymbol{q} = [4, 0]$
5	NIR	GLCM-contrast $\boldsymbol{q} = [4, 0]$
6	Red	GLCM-contrast $\boldsymbol{q} = [16, 0]$
7	Green	GLCM-contrast $\boldsymbol{q} = [8, 0]$
8	Red	GLCM-homogeneity $\boldsymbol{q} = [1, 0]$
9	Green	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
10	Green	GLCM-max(P) $\boldsymbol{q} = [16, 0]$
11	Green	GLCM-max(P) $\boldsymbol{q} = [1, 0]$
12	NIR	GLCM-contrast $\boldsymbol{q} = [4, 4]$
13	NIR	GLCM-homogeneity $\boldsymbol{q} = [0, 4]$
14	Red	wavelet-diagonal $k = 3$
15	Red	wavelet-horizontal $k = 1$
16	Green	GLCM-homogeneity $\boldsymbol{q} = [0, 8]$
17	Green	GLCM-max(P) $\boldsymbol{q} = [8, 0]$
18	Red	GLCM-homogeneity $\boldsymbol{q} = [4, 4]$
19	Red	GLCM-entropy $\boldsymbol{q} = [2, 0]$
20	Green	GLCM-entropy $\boldsymbol{q} = [16, 16]$
21	NIR	GLCM-contrast $\boldsymbol{q} = [2, 0]$
22	NIR	GLCM-correlation $\boldsymbol{q} = [1, 0]$
23	NIR	GLCM-energy $\boldsymbol{q} = [16, 16]$
24	NIR	GLCM-energy $\boldsymbol{q} = [0, 8]$
25	NIR	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
26	Green	GLCM-max(P) $\boldsymbol{q} = [1, 1]$
27	Red	GLCM-entropy $\boldsymbol{q} = [4, 4]$
28	Green	GLCM-homogeneity $\boldsymbol{q} = [16, 0]$
29	Green	GLCM-correlation $\boldsymbol{q} = [0, 8]$
30	Red	GLCM-correlation $\boldsymbol{q} = [0, 16]$

Table 6.1.6: A result from experiment 1. Best spatial features for pine-versus-deciduous using the EQUAL FRACTIONS *set of observations. The mix of second-order texture is similar to when all features where included (see table 6.1.3), except that the Förstner features appears near the bottom of the list.*

rank	channel	feature
1	NIR	GLCM-homogeneity $\boldsymbol{q} = [2, 0]$
2	NIR	GLCM-homogeneity $\boldsymbol{q} = [8, 0]$
3	Green	GLCM-entropy $\boldsymbol{q} = [16, 0]$
4	Red	GLCM-entropy $\boldsymbol{q} = [4, 0]$
5	NIR	GLCM-homogeneity $\boldsymbol{q} = [2, 2]$
6	NIR	GLCM-homogeneity $\boldsymbol{q} = [8, 8]$
7	NIR	GLCM-contrast $\boldsymbol{q} = [8, 0]$
8	Green	GLCM-correlation $\boldsymbol{q} = [0, 1]$
9	NIR	GLCM-correlation $\boldsymbol{q} = [4, 0]$
10	Red	GLCM-homogeneity $\boldsymbol{q} = [1, 0]$
11	NIR	GLCM-contrast $\boldsymbol{q} = [1, 0]$
12	Red	GLCM-homogeneity $\boldsymbol{q} = [0, 4]$
13	Green	GLCM-correlation $\boldsymbol{q} = [0, 16]$
14	Red	GLCM-correlation $\boldsymbol{q} = [0, 8]$
15	Green	GLCM-energy $\boldsymbol{q} = [16, 16]$
16	NIR	GLCM-homogeneity $\boldsymbol{q} = [4, 0]$
17	Green	GLCM-entropy $\boldsymbol{q} = [8, 8]$
18	Red	GLCM-entropy $\boldsymbol{q} = [8, 0]$
19	NIR	wavelet-diagonal $k = 3$
20	Green	wavelet-diagonal $k = 4$
21	Red	wavelet-diagonal $k = 4$
22	NIR	GLCM-correlation $\boldsymbol{q} = [4, 4]$
23	Green	GLCM-contrast $\boldsymbol{q} = [16, 0]$
24	NIR	förstner-normal
25	Red	förstner-normal
26	Green	förstner-tangential
27	Green	GLCM-entropy $\boldsymbol{q} = [16, 16]$
28	Green	wavelet-vertical $k = 4$
29	Red	wavelet-vertical $k = 4$
30	Green	GLCM-homogeneity $\boldsymbol{q} = [4, 4]$

6.2 Experiment 2

For $n' = 30$ best features, the confusion matrices for biased- and unbiased- classification are presented in table 6.2.1. The biased loss (features selected on all data) was $l^*_{30} = 17.6\%$ and the unbiased loss (features selected from training data) was $l_{30} = 23.9\%$. Thus, the bias correspond to of $b_{30} = l_{30} - l^*_{30} = 6.3$ pp. The experiment was also run for the set of $n' = 80$ best-features. The corresponding figures where $l^*_{80} = 14.8\%$ and $l_{80} = 28.1\%$. The biased performance increased, but the unbiased performance degraded so that the bias increased to $b_{80} = 13.3$ pp.

Table 6.2.1: Result of experiment 2. Confusion matrices for classification with $n' = 30$ best features selected from all data (a) and selected from training data (b), respectively. The row-wise class-labels indicate the ground truth class label, and the column-wise class-labels (marked with *) indicate the predicted class-label. The last column shows the fraction of correctly classified observations per class. The bold-font value at the bottom correspond to the total fraction of correctly classified labels. The unbiased classification accuracy is 6.3 pp lower than the biased classification accuracy. Also, the differences in classification accuracy among the different classes increased somewhat.

(a) Biased performance

	spruce*	**pine***	**deciduous***	**% correct**
spruce	227	26	22	82.5
pine	33	226	16	82.2
deciduous	24	24	227	82.5
				82.4

(b) Unbiased performance

	spruce*	**pine***	**deciduous***	**% correct**
spruce	217	26	32	78.9
pine	42	201	32	73.1
deciduous	33	32	210	76.4
				76.1

6.3 Experiment 3

99% (rounded) of the elections were resolved, i.e. not ties. This (rounded) figure did not differ between species.

7 Discussion

Interestingly, the top-features lists (presented in the tables 6.1.1, 6.1.2, and 6.1.3) are dominated by the textural measures based on second-order statistical measures; GLCM and Haar wavelets. This is the case for both the classification results utilizing all features, and for the results utilizing spatial features only (tables 6.1.4, 6.1.5, and 6.1.6). It can also be seen that a wide range of scales and principal directions are utilized by these two approaches to textural features. In fact, for the GLCM features, each step-length and each principal direction occurs at least once on every one of the top-30 lists. The various scales suggests that multiresolution techniques (Mallat, 1989) can effectively describe the tree-crowns in the sense that each scale adds relevant information.

The Förstner features stands out as the only spatial features on the top-30 lists that is not a second-order statistical measure of texture. The success of the Förstner features relative to the other gradient measures is somewhat surprising because the features are derived from the same data, namely the image gradient. A possible explanation is that the Förstner features takes into account the location where each gradient is computed. In contrast, the other gradient features are first-order statistical measures of the gradients that only takes into account what gradients appear in the imagery.

The top-three features of the lists are dominated by spectral measures of the intensity mean. It is therefore surprising to find that the removal of all spectral features from the classification did not impair classification performance substantially. If sufficiently many best-features are used ($n' \geq 35$), the performance gap between the spatial feature subset and the full set of features is negligible (see figures 6.1.3 and 6.1.4). For smaller best-feature sets ($n' < 35$), the penalty from excluding spectral features is modest (see figure 6.1.4), e.g. 2.9 pp at $n' = 20$. This penalty can be dealt with by adding yet more spatial features, i.e. by increasing n'. Another result (see figures 6.1.3 and 6.1.4) is that the removal of all but one spectral channel from the set of spatial features causes a severe drop in classification performance, e.g. utilizing $n' = 20$ features, the performance drop was 13.0 pp for compound classification.

Utilizing the set of $n' = 30$ best-features, the unbiased classification loss of the compound classification is $l_{30} = 23.9\%$, or expressed in classification accuracy 76.1%. This figure is similar to the accuracy of 76.7% achieved in Erikson (2004), and is higher than the accuracy of 67% achieved in Brandtberg (2002).

7.1 Conclusion

The objective of this study is to systemically evaluate a large set of spatial features for tree-crown classification. The result shows that textural measures based on second-order statistics, is the most important class of spatial features among those evaluated in this study. Of these features, both wavelet- and GLCM- features contributes significantly to classification. The ability of wavelets and GLCM to analyse texture at multiple scales is believed to be an important reason of their success.

The objective is also to compare spectral and spatial features, and to investigate what spectral information is necessary. For a small feature set, classification performance ben-

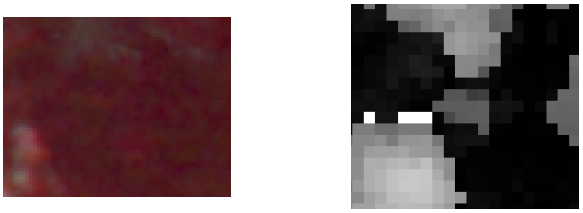
efits from including both spectral and spatial features. However, if sufficiently many features are used, the classification can be conducted based on spatial features only. On the other hand, the multispectral information from utilizing multiple spectral bands is found to be necessary, even for spatial features.

The method of classification turned out to be useful for the particular application. The classification accuracy (76%) is comparable to earlier results. Also, experiment 3 verifies that the voting scheme is able to resolve the large majority of the predictions (99%).

7.2 Limitations

This study has a number of limitations that arise from problems with the dataset utilized. Errors in geo-referencing causes an offset of the trees relative to segments. In some cases, the offset is so large that tree-crowns end up in-between segments. These errors are likely to have a negative impact on classification performance. The extent of in-between segments, and thus their effect on classification performance, is unknown.

A more subtle consequence of using ALS-derived segments is that tree-crowns that would not be detected by an image-based segmentation scheme are included. In particular, trees in the lower canopy that are shadowed by their neighbours may still be distinguishable in the *height model* (see figure 7.2.1). In general, this is believed to have a negative impact on classification performance because image based features cannot be expected to work if the object (in this case tree-crown) cannot be clearly distinguished in the image. Another possibility is that trees in the lower canopy may be particularly likely to belong to a certain species. In real forest stands, it is common that the canopy is layered such that different species dominates different canopy heights. If there is a strong height-species correlation, the classification performance may improve because dark tree-crown-imagery can be assumed to belong to a specific species. This is however not believed to be the case for the dataset utilized in this study, because the trees come from small sampled stands, scattered over a large area.



(a) image *(b) height model*

Figure 7.2.1: An example of a tree that is segmented from the ALS data, but that is unlikely to be segmented by an image based method. The tree in (a) correspond to the small, but clearly distinguishable blob at the centre of the height model in (b). The tree is significantly shorter and smaller than its neighbours, and therefore is shadowed in the optical imagery.

The trees classified in this study are limited to be no farther than 60 meters away from the camera centre, horizontally. For multispectral features, the differences in sun illumination and viewing geometry for trees at different position within the image is a

known problem (Puttonen et al., 2009; Korpela et al., 2011). These differences are also likely to cause similar problems for classification based on spatial features. In a real situation, classification would be conducted on a larger portion of the trees within the image. In such a situation the effect of sun illumination and viewing geometry is more prominent, and consequentially the reported classification accuracy in this study may be overestimated due to this distance limit.

7.3 Future work

To determine if the second-order textural measures are practical for real forest inventory applications, the next step is to investigate what classification performance can be achieved from the features. Firstly, the method should be evaluated using an image based segmentation scheme (as would be used in practice), e.g. the one utilized in Erikson (2004), or one from Brandtberg and Warner (2006). Furthermore, because the number of features utilized in this study is large and because the effect of overfitting is believed to be more prominent than seen figure 6.1.2, it is feasible to apply methods of *dimensionality reduction* (decreasing the number of dimensions in the feature vector) in excess off the feature selection. A simple approach that has been utilized in tree-crown classification literature is *principal component analysis* (e.g. in Meyera et al., 1996; Brandtberg, 2002; Ørka et al., 2007).

The optimal number of features to use could not be estimated, because of the bias problem. It is therefore a topic for future work to implement a version of experiment 1 that computes the unbiased losses.

Important feature classes that were excluded from this study were *geometric features*, i.e. features that describe the shape, area or boundary of the tree crown, or regions within the tree-crown. Because this class of features is believed to add non-redundant information over the second-order texture, it should be included in future work. This set of features was excluded from this study on the basis that the shape of the ALS-derived segments does not correspond to the shapes of the image-derived segments that would be used in a real situation. In tree-crown classification literature, examples of work that utilizes geometric features are found in Brandtberg (1998, 2002); Erikson (2004).

In this study, performance was evaluated in terms of *classification accuracy*. Another kind of performance that is of high importance for practical applications is *robustness*. A robust tree-crown classifier is expected to work under different environmental conditions affecting the dataset (e.g. in summer and in fall), and to tolerate variations within the same dataset. A particularly important problem with multispectral features is that differences in sun-illumination and viewing geometry at different locations within the image cause tree-crowns to be spectrally different from each other (Puttonen et al., 2009; Korpela et al., 2011). In section 1.2, this was presented as a reason to switch from multispectral to spatial features. However, the claim that spatial features are more robust to these two factors must be confirmed by experiments. A topic for future work is therefore to include tree-crowns at more various positions relative to the camera centre. In particular tree-crowns that are farther away than 60 meter.

8 Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Niclas Börlin, for his patience, expertise, and support during these five months. His guidance, sharp eye for detail, and inexhaustible supply of ideas, helped me from the beginning of this project to the very end.

Besides my advisor, I would also like to thank Mattias Nyström and Jonas Bohlin for their enthusiasm and sincere interest, and for aiding me with their expertise in the field of forest science. I thank them for suggesting the project in the first place.

I would also like to thank Johan Holmgren whom provided me with the tree-crown segments vital to this work.

Last but not the least, I would like to thank my family; my mother and father for raising me, and for supporting me throughout my life.

References

Aggarwal, N. and Agrawal, R. (2012). First and second order statistics features for classification of magnetic resonance brain images.

Alm, S. E. and Britton, T. (2008). *Stokastik: sannolikhetsteori och statistikteori med tillämpningar*. Liber.

Boman, J. (2013). Tree species classicationusing terrestrial photogrammetry.

Brandtberg, T. (1997). Towards structure-based classification of tree crowns in high spatial resolution aerial images. *Scandinavian Journal of Forest Research*, 12(1):89–96.

Brandtberg, T. (1998). Algorithms for structure-and contour-based tree species classification using digital image analysis. In *Proceedings of the International Forum on Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry*, pages 199–207.

Brandtberg, T. (2002). Individual tree-based species classification in high spatial resolution aerial images of forests using fuzzy sets. *Fuzzy Sets and Systems*, 132(3):371–387.

Brandtberg, T. (2007). Classifying individual tree species under leaf-off and leaf-on conditions using airborne lidar. *ISPRS Journal of Photogrammetry and Remote Sensing*, 61(5):325–340.

Brandtberg, T. and Warner, T. (2006). High-spatial-resolution remote sensing. In *Computer applications in sustainable forest management*, pages 19–41. Springer.

Brandtberg, T., Warner, T. A., Landenberger, R. E., and McGraw, J. B. (2003). Detection and analysis of individual leaf-off tree crowns in small footprint, high sampling density lidar data from the eastern deciduous forest in north america. *Remote sensing of Environment*, 85(3):290–303.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167.

Erikson, M. (2004). Species classification of individually segmented tree crowns in high-resolution aerial images using radiometric and morphologic image measures. *Remote Sensing of Environment*, 91(3):469–477.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Förstner, W. (1986). A feature based correspondence algorithm for image matching. *International Archives of Photogrammetry and Remote Sensing*, 26(3):150–166.

Förstner, W. and Gülch, E. (1987). A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, pages 281–305.

Fournier, R. A., Edwards, G., and Eldridge, N. R. (1995). A catalogue of potential spatial discriminators for high spatial resolution digital images of individual crowns. *Canadian Journal of Remote Sensing*, 21(3):285–298.

Franklin, S., Hall, R., Moskal, L., Maudie, A., and Lavigne, M. (2000). Incorporating texture into classification of forest species composition from airborne multispectral images. *International Journal of Remote Sensing*, 21(1):61–79.

Franklin, S., Maudie, A., Lavigne, M., et al. (2001). Using spatial co-occurrence texture to increase forest structure and species composition classification accuracy. *Photogrammetric Engineering and Remote Sensing*, 67(7):849–856.

Gonzalez, Rafael C; Richard, E. W. (2008). *Digital image processing*. Prentice Hall, 3'd edition.

Gougeon, F., Moore, T., et al. (1989). Classification individuelle des arbres à partir d'images à haute résolution spatiale.

Gougeon, F. A. (1995). Comparison of possible multispectral classification schemes for tree crowns individually delineatedon high spatial resolution meis images. *Canadian Journal of Remote Sensing*, 21(1):1–9.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422.

Guyot, G., Guyon, D., and Riom, J. (1989). Factors affecting the spectral response of forest canopies: a review. *Geocarto International*, 4(3):3–18.

Haar, A. (1910). Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen*, 69(3):331–371.

Haralick, R. M., Shanmugam, K., and Dinstein, I. H. (1973). Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):610–621.

Holmgren, J. and Lindberg, E. (2013). Tree crown segmentation based on a geometric tree crown model for prediction of forest variables. *Canadian Journal of Remote Sensing*, 39(sup1):S86–S98.

Holmgren, J., Persson, Å., and Söderman, U. (2008). Species identification of individual trees by combining high resolution lidar data with multi-spectral images. *International Journal of Remote Sensing*, 29(5):1537–1552.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425.

Kaartinen, H., Hyyppä, J., Yu, X., Vastaranta, M., Hyyppä, H., Kukko, A., Holopainen, M., Heipke, C., Hirschmugl, M., Morsdorf, F., et al. (2012). An international comparison of individual tree detection and extraction using airborne laser scanning. *Remote Sensing*, 4(4):950–974.

Ke, Y., Quackenbush, L. J., and Im, J. (2010). Synergistic use of quickbird multispectral imagery and lidar data for object-based forest species classification. *Remote Sensing of Environment*, 114(6):1141–1154.

Key, T., Warner, T. A., McGraw, J. B., and Fajvan, M. A. (2001). A comparison of multispectral and multitemporal information in high spatial resolution imagery for classification of individual tree species in a temperate hardwood forest. *Remote Sensing of Environment*, 75(1):100–112.

Korpela, I., Heikkinen, V., Honkavaara, E., Rohrbach, F., and Tokola, T. (2011). Variation and directional anisotropy of reflectance at the crown scale—implications for tree species classification in digital aerial images. *Remote Sensing of Environment*, 115(8):2062–2074.

Korpela, I., Ørka, H. O., Maltamo, M., Tokola, T., Hyyppä, J., et al. (2010). Tree species classification using airborne lidar—effects of stand and tree parameters, downsizing of training set, intensity normalization, and sensor type. *Silva Fennica*, 44(2):319–339.

Leckie, D., Gougeon, F., Hill, D., Quinn, R., Armstrong, L., and Shreenan, R. (2003a). Combined high-density lidar and multispectral imagery for individual tree crown analysis. *Canadian Journal of Remote Sensing*, 29(5):633–649.

Leckie, D. G., Gougeon, F. A., Walsworth, N., and Paradine, D. (2003b). Stand delineation and composition estimation using semi-automated individual tree crown analysis. *Remote Sensing of Environment*, 85(3):355–369.

Leckie, D. G., Tinis, S., Nelson, T., Burnett, C., Gougeon, F. A., Cloney, E., and Paradine, D. (2005). Issues in species classification of trees in old growth conifer stands. *Canadian Journal of Remote Sensing*, 31(2):175–190.

Li, J., Hu, B., Sohn, G., and Jing, L. (2010). Individual tree species classification using structure features from high density airborne lidar data. In *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*, pages 2099–2102. IEEE.

Lillesand, T. M., Kiefer, R. W., Chipman, J. W., et al. (2004). *Remote sensing and image interpretation.* John Wiley & Sons Ltd, 5'th edition.

Mallat, S. G. (1989). A theory for multiresolution signal decomposition: the wavelet representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(7):674–693.

Meyera, P., Staenzb, K., and Ittena, K. (1996). Semi-automated procedures for tree species identification in high spatial resolution data from digitized colour infrared-aerial photography. *ISPRS Journal of Photogrammetry and Remote Sensing*, 51(1):5–16.

Nordkvist, K. and Olsson, H. (2013). Laserskanning och digital fotogrammetri i skogsbruket.

Ørka, H., Næsset, E., and Bollandsås, O. (2007). Utilizing airborne laser intensity for tree species classification. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(Part 3):W52.

Ørka, H. O., Næsset, E., and Bollandsås, O. M. (2009). Classifying species of individual trees by intensity and structure features derived from airborne laser scanner data. *Remote Sensing of Environment*, 113(6):1163–1174.

Packalén, P., Suvanto, A., and Maltamo, M. (2009). A two stage method to estimate species-specific growing stock. *Photogrammetric Engineering & Remote Sensing*, 75(12):1451–1460.

Persson, A., Holmgren, J., and Söderman, U. (2002). Detecting and measuring individual trees using an airborne laser scanner. *Photogrammetric Engineering and Remote Sensing*, 68(9):925–932.

Pinz, A., Zaremba, M. B., Bischof, H., Gougeon, F. A., Locas, M., et al. (1993). Neuromorphic methods for recognition of compact image objects. *Machine Graphics and Vision*, 2(3):209–229.

Pinz, A. J. and Bischof, H. (1990). Constructing a neural network for the interpretation of the species of trees in aerial photographs. In *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, volume 1, pages 755–757. IEEE.

Puttonen, E., Litkey, P., and Hyyppä, J. (2009). Individual tree species classification by illuminated—shaded area separation. *Remote Sensing*, 2(1):19–35.

Ramsey, J. B., Newton, H. J., and Harvill, J. L. (2002). *The elements of statistics: With applications to economics and the social sciences*. Duxbury/Thomson Learning.

Russell, S. and Norvig, P. (2010). *Artificial intelligence: A modern approach*. Prentice Hall, 3'rd edition.

Shao, G. and Reynolds, K. M. (2006). *Computer applications in sustainable forest management: Including perspectives on collaboration and integration*, volume 11. Springer Science & Business Media.

Strang, G. (2009). *Introduction to linear algebra*. Wellesley-Cambridge Press, Massachusetts, 4'th edition.

Tang, J., Alelyani, S., and Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications. Editor: Charu Aggarwal, CRC Press In Chapman & Hall/CRC Data Mining and Knowledge Discovery Series*.

Vauhkonen, J., Ene, L., Gupta, S., Heinzel, J., Holmgren, J., Pitkänen, J., Solberg, S., Wang, Y., Weinacker, H., Hauglin, K. M., et al. (2011). Comparative testing of single-tree detection algorithms under different types of forest. *Forestry*, page cpr051.

Walter, F. (1998). Fjärranalys för skoglig planering.

West, P. W. and West, P. W. (2009). *Tree and forest measurement*. Springer.