Single image focus level assessment using Support Vector Machines

Oscar Beijbom

30 January 2007

Abstract

Differential white blood cell count is the process of counting and classifying white blood cells in blood smears. It is one of the most common clinical tests which is performed in order to make diagnoses in conjunction with medical examinations. These tests indicate deceases such as infections, allergies, and blood cancer and approximately 200-300 million are done yearly around the world.

Cellavision AB has developed machines that automate this work and is the global leader in this market. The method developed in this thesis will replace and improve the auto focus routine in these machines. It makes it possible to capture a focused image in only two steps instead of using an iterative multi step algorithm like those used today in most auto focus systems, including the one currently used at Cellavision.

In the proposed method a Support Vector Machine, SVM, is trained to assess quantitatively, from a singel image, the level of defocus as well as the direction of defocus for that image. The SVM is trained on features that measure both the image contrast and the image content. High precision is made possible through extracting features from the different parts of the image as well as from the image as a whole. This requires the image to be segmented and a method for doing this is proposed.

Using this method 99.5% of the images in the test data's distances to focus were classified less or equal to $5\mu m$ wrong while over 85% were classified completely correctly. A $5\mu m$ defocus is borderline to what the human eye perceives as defocused.

Cellavision AB has applied for a patent to protect the method described in this thesis.

Acknowledgements

I would like to thank my supervisors, Professor Karl Åström and Anders P Eriksson at the department of mathematics, LTH, and Sven Hedlund and Kent Stråhlen at Cellavision AB for a great support throughout my work. Thank you for all your time and help! In particular you, Sven, for the many hours spent discussing solutions and steps ahead, thanks! Also, brothers in arms, Per Sennmalm and Adam Morell from Cellavision, thank you both for valuable input on various image analysis topics.

Contents

Chapter 1

Introduction

This master thesis describes how an absolute focus metric can be created in images of stained peripheral blood smears using multi variable analysis.

The author is following the Master of Science in Engineering Physics program at Lund Institute of Technology, LTH, in Sweden. He has focused his studies on image analysis and computer vision. The master thesis was done at a company in Lund, Cellavision AB, in collaboration with the Center of Mathematical Sciences at LTH.

Cellavision AB, www.cellavision.com, is developing, marketing and selling equipment for Automated Digital Cell Morphology. This is the process by which white blood cells, WBCs, are automatically located in a stained peripheral blood smear, pre-classified, stored and transmitted for confirmation by a technologist. Cellavision was the first company in this market, and is still today the market leader.

1.1 Equipment and definitions

Fig. 1.1 shows a schematic image of the equipment used. Let z be the distance between object and lens, and $[x, y]$ the location in the plane perpendicular to z. The microscope is movable in the z direction while the slideholder moves in the x and y directions. Let f be the distance so that when $z = f$ the image is focused. If $z > f$ the image is positively defocused and if $z < f$ the image is negatively defocused. The distance, f, is typically around $0.1mm = 100\mu m$. If the distance $|z - f|$ is larger than $0.5\mu m$ the images are perceived as defocused for the human eye. The system has been built from ground up in Matlab 7.1 running on Pentium(R) 4 CPU 3.00GHz PC with 496 MB RAM. Toolboxes 'Oregon State University Support Vector Machine Toolbox' from Oregon State University, and the 'Wavelab 850'

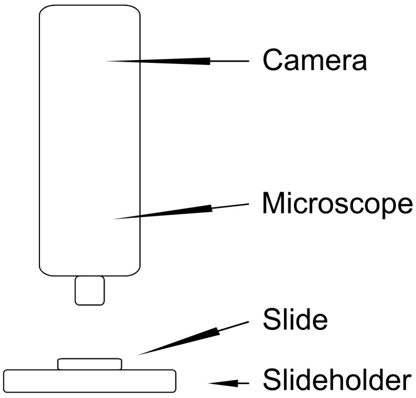

Figure 1.1: schematic view of the microscope and object

from Stanford University have been downloaded and used. The RGB-color space has been used throughout the thesis.

1.2 Background and usefulness of an absolute focus metric

The process of counting and classifying white blood cells in a blood smear is called 'differential white blood cell count'. It is one of the most common clinical tests which is performed in order to make diagnoses in conjunction with medical examinations. These tests indicate deceases such as infections, allergies, and blood cancer and approximately 200-300 million are done yearly around the world.

Cellavision AB has developed two products, the DM8 and the DM96, for automation of differential white blood cell counts. These machines standardize the analysis, which gives a more objective assessment of the test. In addition, this enables training and skills enhancement, digital archiving of tests together with patient files, and transfer of digital graphics to experts outside the laboratory. Cellavision is the first and biggest company in this market and competes mainly with manual microscope analysis. Images from both machines are used in this thesis.

These machines work in the following way. First, the operator loads the machine with slides. The slides are thin rectangular pieces of glass on witch a stained blood sample has been smeared. In the DM96 up to eight magazines holding 12 slides each can be loaded, hence the ”96” in DM96. These slides are then lifted one by one by the machine onto the slideholder, which is the table that is movable in the horizontal plane. In the DM8, eight slides are mounted manually onto the slideholder. The slideholder is bigger in the DM8 than in the DM96 and there are some indications that this causes trouble for the auto focus method used today due to more vibrations and lower precision.

Once the slides are mounted the machines work in the same way. First the area of the smear with the right thickness is found. Then an image with lower magnification (10X in the DM96 and 50X in the DM8) is taken and the locations of the WBCs are found. Then the microscope switches to a higher magnification (100X) and the slide holder places the WBCs one by one under the camera. A focused image is thus captured and sent to the classifier for the pre-classification. The focused images and the pre-classifications are stored in a database and the operator can go through the material, verify the pre-classification and look at the images.

Now, let's look closely at how the machines captures a focused image, and how this procedure can be improved using an absolute focus metric.

Capturing a focused image today involves capturing several images, up to 20, at different distances, z, and comparing the amount of high frequency components in these images. The image with the highest amount of high frequency components is selected as focused. Since the blood smear appears to be a plane object and the slide holder is horizontal one could think that finding focus is required only once and that the same height of the microscope can be used for the rest of the WBCs. This, in fact, is not the case. Firstly, the blood smear isn't horizontal but in fact slopes slightly. In fact it is not even fully planar. Secondly, plays in the mechanism that moves the table and vibrations caused by the engines moving the table makes it hard for the controlling algorithm to estimate the exact distance between the table and the microscope. Also, the depth of focus is shallow due to the strong magnification used which complicates matters further. Still, it is possible to make a prediction of the z value needed for given x and y values after having captured a number of images. Today, however, there is no way to know if the prediction is correct and additional images have to be captured even it in fact was correct. The method used today works but leaves space for improvement.

An absolute focus metric could speed up the focus routine in several ways. The most basic is to have a yes/no function that would evaluate if the predictions are correct, and if they are not, let the auto focus routine work its way to a focused image the same way it does today. A more advanced use of the absolute focus metric requires that the metric indicates not only if the image is defocused but also how defocused it is and in what direction. With this information the auto focus routine could work in two steps. If the first prediction turns out to be defocused, the microscope simply moves the distance given by the metric and the second captured image is focused. Also in this case there might be a need for a supporting algorithm to deal with abnormal situations.

1.3 Problem description

The purpose of this master thesis is to evaluate to which extent it is possible to create an absolute focus metric in blood smear images using multi variable analysis. This is to be done in two steps. First determine if the computer can decide if an image is focused or not without taking into consideration the level of defocus. Then, if this works, develop a system that determines quantitatively how defocused an image is. Incorporated in this second step will be to determine if an image is negatively or positively defocused. I will refer to the first part of the problem as the "two class problem" and the second part as the "multi class problem".

1.4 Problem analysis and prerequisites

The basic idea is fairly simple; identify and extract relevant features and then train a classifier. The features will be divided in three groups. One group that measures the amount of high level frequencies in the images, one that indicates if the image is positively or negatively defocused, and one that characterizes the content of the image. The ambition is that the third group of features will help the classifier to, in some sense, normalize the measurements from the first and second group. Fig. 1.2 shows an overview of the proposed method.

Generally speaking a focused image contains a higher amount of high frequencies than a defocused one. However, when measuring the amount of high frequencies in an image, several factors will have an impact; the focus level, the amount of edges in the image, the color intensities, variances, and more. A focused image of a pale and smooth object can contain less high frequency than a slightly defocused image of a colorful object with many details.

A basic prerequisite for this thesis is therefore the similarity of the images analyzed. They usually contain one or more white blood cells, **WBCs**, and always red blood cells, **RBCs**. However, there are several things that compromise this statement. The following is an attempt to summarize the variations that exist in the images used for training and testing.

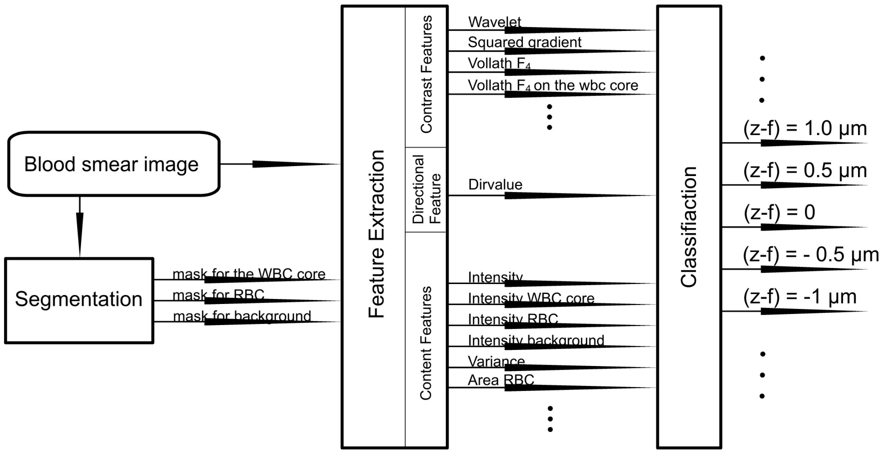

Figure 1.2: Overview of the work done in this thesis

- The stain method. The images are usually stained either with a method called May-Grünwald-Giemsa, MGG, which is the preferred method in Europe, or with Wright, which is more common in USA. Different stain methods give different colors in the image.
- The level of stain. A poorly stained image is less colorful than a heavily stained one.
- The smear method. Some labs use a machine to smear the blood onto the slides while other do it manually. A manually smeared blood sample is not as smooth as one smeared by a machine.
- Dirt. The images contain different amounts of dirt.
- Variations in thickness. The layer of RBCs is thicker in some images and the RBCs are on top of each other. This makes the edges look blurry.
- Different amount of WBCs. In some cases the image lacks a WBC, due to failure in the algorithm that locates them, and in some cases two WBCs are so close that both appear in the same image.
- The number of RBCs.

For a classifier to determine the focus level from only one image it must be given features that somehow measure the variations above. Fig. 1.3 shows four examples of how the focused images from the data can look. The WBC is the dark, big object in the center of the images, and the RBCs are the smaller brighter round objects. Note the large differences in the amount of RBCs, the color of the RBCs and the shapes of the WBCs. Fig. 1.4

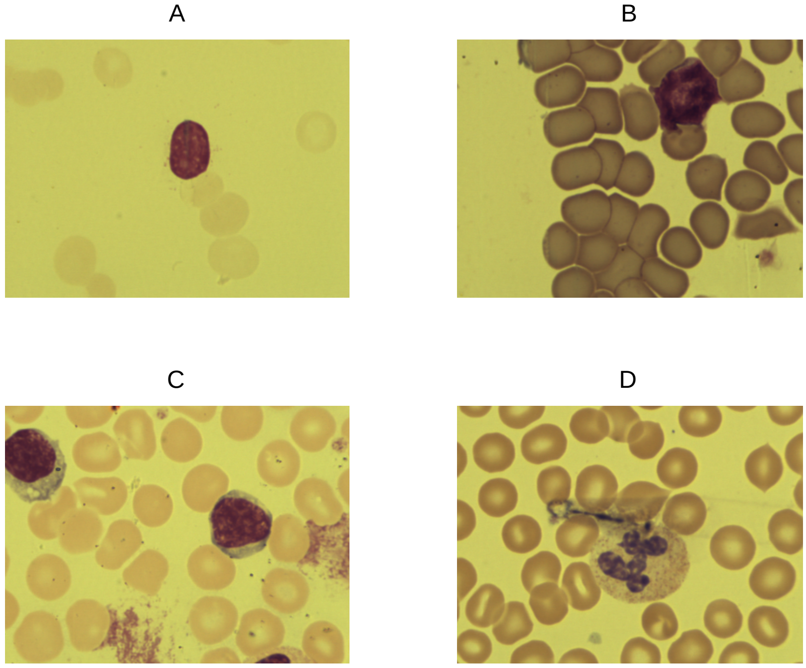

Figure 1.3: Four examples of images in the materials used. The WBC is the dark, big object in the center of the images, and the RBCs are the smaller brighter round objects. Note the big differences in the amount of RBCs, the color of the RBCs and the shapes of the WBCs.

shows image D from Fig. 1.3 at six different level of defocus $(z - f) = [45, 15, 10, 5, -5, -45]\mu m$. Note how the WBC in the image with $(z - f) = 0.5$ micrometer is better focused than the WBC in the image with $(z - f) = -0.5$. The reason for this is explained below.

When determining if the image is negatively or positively focused there are at least two different facts that can be exploited. First is the fact that a WBC is larger than a RBC. The images taken at a positive defocus can thus still be focused on the WBC while blurry on the rest of the image. The images captured at negative defocus will have a blurrier WBC than positive defocused images captured at the same distance from focus. This is illustrated in Fig. 1.5 and is also visible to some extent in Fig. 1.4. The series of images used to create the graphs in Fig. 1.5 and also in Fig. 1.6 is different from the ones used in the rest of this thesis in that the distance z is only changed $0.1\mu m$ between the images. The image with the highest global contrast in the green color layer is selected as the best focused image, and f is set so that $z - f = 0$ for this image. On the y-axis is the image

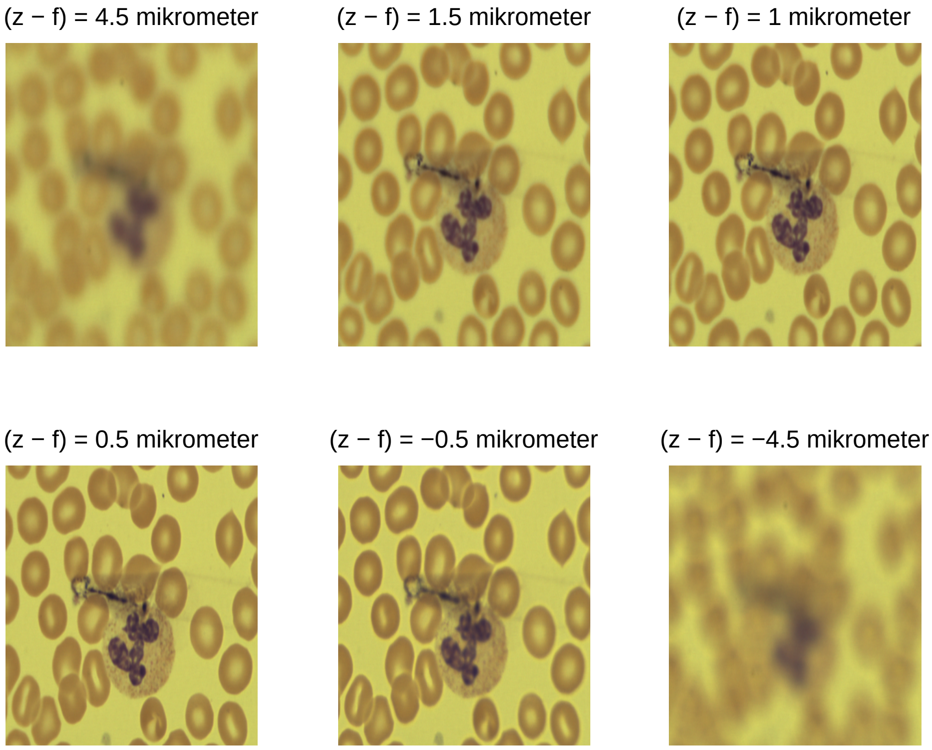

Figure 1.4: Image D from Fig. 1.3 at six different levels of defocus $(z - f) = [45, 15, 10, 5, -5, -45]\mu m$*. Note how the WBC in the image with* $(z - f) = 0.5\mu m$ *is better focused than the WBC in the image with* $(z - f) = -0.5\mu m$*.*

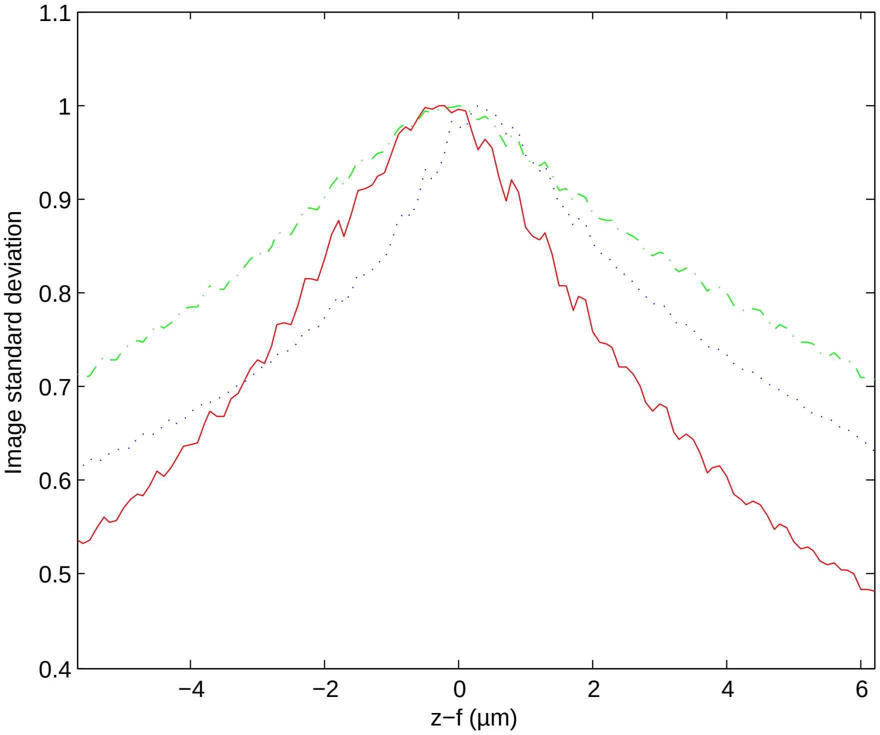

Figure 1.5: Standard deviation for the green color layer at the WBC (blue dotted line), the whole image (green dash-dot line), and the image minus the WBC (red solid line). The size of the WBC makes it focused on a different distance, z, than the rest of the image.

standard deviation calculated as

$$STD = \sqrt{\sum_{allpixels} (pixel\ value - mean\ pixel\ value)^2}.$$

This is a basic way to measure the level of focus. The pixel intensities in a focused image will vary more than in a defocused one. Only the green color layer is used in Fig. 1.5. The blue dotted line, with its highest value for $x = 0.3$, represents the STD for the WBC. The green dash-dot line, with its peak at $x = 0$ the STD for the whole image and the red solid line, with its highest value for $x = -0.2$, the STD for the part of the image that does not contain the WBC. This confirms that the height of the WBC indeed makes it focused on a different distance than the rest of the image, and it is something that can be used to determine the direction of defocus.

The second fact that can be used to determine if an image is negatively or

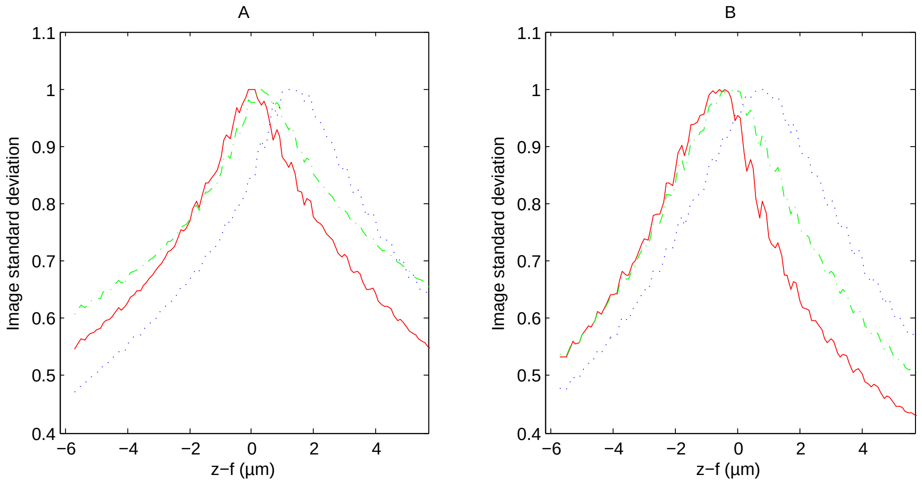

Figure 1.6: The different color layers of an image are focused at different distances. Image A illustrates the three color layers for only the WBC and image B illustrates the three color layers for the whole image minus the WBC. The red solid line represents the red color layer, the green dash-dot represents the green color layer and the blue dotted line the blue layer. Note that the difference between the red peak and the blue peak is quite large, 1.4μm. *Also note that the tails in the positive and negative directions of defocus are different from each other, and also differ between images A and B.*

positively defocused is the following. Different colors captured by the camera have different wavelengths and will be focused at different distances between object and lens. This is compensated for in some lens systems but not fully in the one used at Cellavision. Fig. 1.6 illustrates this phenomenon. The same images are used as in Fig. 1.5, and the same measurement of focus level. Here the three graphs illustrate the different color layers. The red solid line represents the red color layer, the green dash-dot represents the green color layer and the blue dotted line the blue layer. To avoid influence from the effects illustrated in Fig. 1.5 this test is done two times. Image A illustrates the three color layers for only the WBC and image B illustrates the three color layers for the whole image minus the WBC. Note that the difference between the red peak and the blue peak is quite large, 1.4μm. Also note that the tails in the positive and negative directions of defocus are different from each other, and also differ between images A and B. All this is information that hopefully can be used by the SVM when separating the direction of defocus.

1.5 Literature review

The issue of auto focus has been covered in numerous scientific articles and publications. Several relative focus metrics are proposed and combined with different iterative algorithms. Less is done in the field of absolute focus metrics. During an initial search very little was found. The closest was in [1] where a wavelet based absolute measurement is proposed. However, tests revealed that this did not work for the images that are worked on in this thesis. In section 3.1.1 is an overview and evaluation of different relative focus metrics.

As the results of this thesis got better a discussion on applying for a patent began, and a search was conducted to find if there were any existing conflicting patents. As it turned out, there was a patent application [13] filed September 14 2006 in the United States that put forward a similar idea. It does not, however, aspire to solve the multi class problem but only the two class problem, and a patent application is now filed by Cellavision AB to protect the idea presented in this thesis.

Chapter 2

Image processing preliminaries

This chapter explains the theory behind some of the methods used in the thesis. The part on Support Vector Machines, SVM, is thorough as this is a central part of the thesis. The other methods are more briefly described.

2.1 Support vector machines

Support Vector Machines were introduced in COLT, Conference on Computational Learning Theory, 1992 by Boser Guyon, Vapnik. It is a classifier that is used in similar applications as Artificial Neural Networks but that, in addition to giving as satisfactory results as ANN in most applications, has more appealing theoretical properties. Below follows an introduction to SVM classifiers. For a more thorough description cf. [5].

2.1.1 Linearly separable datasets

Consider the simplest possible classification problem; one with two linearly separable classes. Formally, we want to estimate a function $f : \Re^N \to \{\pm 1\}$ using data

$$(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_l}, y_l) \in \Re^N \times \{\pm 1\}.$$

With $\mathbf{x_i}$ and y_i as above there exists a hyperplane such as

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b > 0. \tag{2.1}$$

Points that lie "below" the plane i.e. points for which $(\mathbf{w} \cdot \mathbf{x_i} + b) < 0$ belong to $y_i = -1$ and are thus multiplied with -1 making the equation (2.1) true

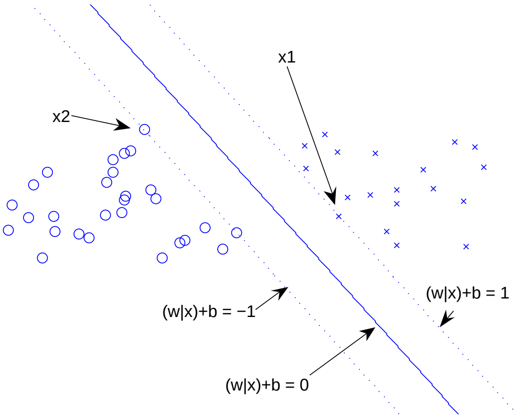

Figure 2.1: A linearly separable set of data and the separating hyper plane

for all points. Unknown points can now be classified as

$$\mathbf{f}(\mathbf{x}) \quad = \quad sgn((\mathbf{w} \cdot \mathbf{x}) + b). \tag{2.2}$$

In a set with a finite number of data points there are several (infinite) ways of choosing $\mathbf{w}$ and b. The most intuitive way is to choose the plane that is furthest away from the convex hull of the two sets of data points. This is obtained through joining the closest points on the two hulls with a line and let the hyperplane be perpendicular to this line and cut it midway. Through rescaling $\mathbf{w}$ and b in (2.1) so that the points closest to the hyper plain satisfy $\|(\mathbf{w} \cdot \mathbf{x}) + b\| = 1$, we obtain a form $(\mathbf{w}, b)$ of the hyper plane, so that

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b \geq 1$$

for all points with equality for the point(s) from both sets that are closest to the hyper plane. These points are called the support vectors. Fig. 2.1 shows the situation dealt with in this section; a linearly separable set of data and its separating hyper plane. The support vectors are the points that lie on either of the dashed lines. Let $\mathbf{x_1}$ be a support vector from class one and $\mathbf{x_2}$ one from class two. Then

$$\begin{aligned} (\mathbf{w} \cdot \mathbf{x_1}) + b &= 1, \\ (\mathbf{w} \cdot \mathbf{x_2}) + b &= -1, \\ (\mathbf{w} \cdot (\mathbf{x_1} - \mathbf{x_2})) &= 2. \end{aligned} \tag{2.3}$$

Now consider the distance from $\mathbf{x_1}$ to $\mathbf{x_2}$ in the direction of the hyperplane normal, $\mathbf{w}/\|\mathbf{w}\|$,

$$\frac{(\mathbf{w} \cdot (\mathbf{x_1} - \mathbf{x_2}))}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|},$$

where the equality is from equation (2.3). Maximizing the distance from the hyper plane to the points thus means minimizing $\|\mathbf{w}\|$, which in turn is equal to minimizing $\|\mathbf{w}\|^2$ Now we can formulate the problem that needs to be solved to find the desired hyper plane,

$$\begin{array}{rcr} \min & \|\mathbf{w}\|^2 & (2.4) \\ \text{subject to} & y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1, \; i = 1, \ldots, l. & (2.5) \end{array}$$

The following is an outline on how to solve this kind of problem. It is a standard optimization problem known as quadratic programming, and can be solved using standard methods. These problems are convex, which means that there exist no local minima other than the global minima. The optimization problems generated using ANN does not have this property and this is the reason behind the 'more appealing theoretical properties' mentioned in the introduction to this section.

Before solving it we will transform it to its dual counterpart. For the theory behind this transformation and the details on how to solve a quadratic programming problem please refer to [3].

Introduce Lagrange multipliers $\alpha_i \geq 0$ and the Lagrangian

$$L(\mathbf{w}, b, \alpha) \;=\; \|\mathbf{w}\| - \sum_{i}^{l} \alpha_i \; (y_i \cdot ((\mathbf{x_i} \cdot \mathbf{w}) + b) - 1).$$

The Lagrangian L has to be minimized with respect to the primal variables $\mathbf{w}$ and b and maximized with respect to the dual variables α_i These are the Karush-Kuhn-Tucker conditions described in [3]. According to these the derivatives of L with respect to the primal variables must be zero,

$$\frac{\partial}{\partial b} L(\mathbf{w}, b, \alpha) = 0, \; \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \alpha) = 0,$$

which leads to

$$\sum_{i=1}^{l} \alpha_i y_i \;=\; 0, \qquad (2.6)$$

$$\mathbf{w} \;=\; \sum_{i=1}^{l} \alpha_i y_i \mathbf{x_i}. \qquad (2.7)$$

The solution vector $\mathbf{w}$ can thus be written as a sum of a subset of the training vectors, namely the support vectors. By the KKT conditions all non-active constraints have $\alpha_i = 0$ and do not affect the position of the separating hyper plane. This fits with the idea that only the vectors closest to the hyper plane affect it.

By substituting (2.6) and (2.7) into L, the primary variables are eliminated and the dual, which is also a quadratic programming problem, becomes

$$\max \quad \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x_i} \cdot \mathbf{x_j}), \tag{2.8}$$

$$\text{subject to} \quad \alpha_i \geq 0,\ i = 1, \ldots, l,\ \textit{and} \sum_{i=1}^{l} \alpha_i y_i = 0. \tag{2.9}$$

Solving it gives $\alpha_i, i = 1, \ldots, l$. The vector $\mathbf{w}$ is then given by (2.7). For any example $\mathbf{x_i}$ with α_i nonzero,

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b = 1,$$

which can be used to calculate b. The decision function (2.2) for an unknown point $\mathbf{x_u}$ becomes

$$\mathbf{f}(\mathbf{x}) \quad = \quad sgn(\sum_{i=1}^{l} \alpha_i y_i (\mathbf{x_u} \cdot \mathbf{x_i}) + b). \tag{2.10}$$

Note how the data points occur only within the scalar product in (2.10) and (2.8) while it doesn't in (2.4). This is the reason the transformation was made and why this is important will become evident in Section 2.1.2.

2.1.2 Non-linearly separable data sets

Building on the same model, we now want to be able to work with non-linearly separable data sets. The idea is to map the data points with a non-linear function into a feature space where they will be linearly separable, cf. Fig. 2.2. This feature space will have a higher dimension. This mapping is done through a function $\phi(\mathbf{x}) : \mathcal{R}^N \rightarrow \mathcal{F}$. We will see that it is useful to use another function called the Kernel function which is defined as

$$k(\mathbf{x_i}, \mathbf{x_j}) \quad := \quad (\phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})). \tag{2.11}$$

Let's look at an example before we proceed. The polynomial kernel of dimension 2

$$k(\mathbf{x}, \mathbf{y}) \quad = \quad \gamma(\mathbf{x} \cdot \mathbf{y})^2, \tag{2.12}$$

can be shown to correspond to a map Φ into the space spanned by all products of exactly 2 dimensions. For $\mathbf{x} = (x_1, x_2), \mathbf{y} = (y_1, y_2), \gamma = 1$ this

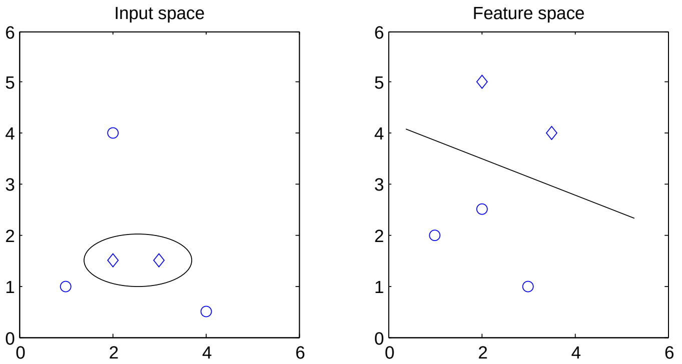

Figure 2.2: Use a Kernel to map the training data into a higher dimension where it will be linearly separable.

becomes

$$\begin{aligned}(\mathbf{x}\cdot\mathbf{y})^2 &= (x_1y_1+x_2y_2)^2 = \\ &= x_1^2y_1^2+x_2^2y_2^2+2x_1y_1x_2y_2 = \\ &= ((x_1^2,x_2^2,\sqrt{2}x_1x_2)\cdot(y_1^2,y_2^2,\sqrt{2}y_1y_2)) = \\ &= (\phi(\mathbf{x})\cdot\phi(\mathbf{y})),\end{aligned}$$

with $\phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$. This works the same way for higher dimensional polynomial kernels. For data with dimension 65 and a polynomial kernel of dimension 3, this would result in a $(65+3-1)!/(3!*(65-1)!) \approx 48.000$ dimensional feature space.

As previously noted the functions that are to be optimized contain only the scalar products between the data points rather than the points themselves. This means that the function $\phi(x)$ doesn't have to be explicitly defined or used. Instead the kernel is used in all calculations. The transformed data does not have to be stored and the high dimension scalar multiplication does not have to be performed, as the Kernel function does these steps in one. As the example above shows this saves computational power and memory and is in some sense the essence of the SVM theory.

The decision function (2.10) becomes

$$\begin{aligned}\mathbf{f}(\mathbf{x}) &= sgn(\sum_{i=1}^{l} \alpha_i y_i (\phi(\mathbf{x_u}) \cdot \phi(\mathbf{x_i})) + b) \\ &= sgn(\sum_{i=1}^{l} \alpha_i y_i k(\mathbf{x_u}, \mathbf{x_i}) + b) \qquad (2.13)\end{aligned}$$

and the dual optimization (2.8) becomes

$$\max \qquad \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j k(\mathbf{x_i}, \mathbf{x_j}) \qquad (2.14)$$

$$\text{subject to} \quad \alpha_i \geq 0,\ i = 1, \ldots, l, \text{and} \sum_{i=1}^{l} \alpha_i y_i = 0, \qquad (2.15)$$

where b can be calculated through selecting a $\mathbf{x_j}$ with $\alpha_j \neq 0$ using

$$b = y_j - (\sum_{i=1}^{l} \alpha_i y_i k(\mathbf{x_j}, \mathbf{x_i})). \qquad (2.16)$$

2.1.3 Non separable data sets

Now we can handle non-linear separable data. Let's look at the final problem, how to deal with data sets that are non-separable, even after mapping the data into a higher dimensional feature space. To do this introduce slack variables

$$\xi_i \geq 0,\ i = 1, \ldots, l, \qquad (2.17)$$

together with relaxed constraints

$$y_i(\mathbf{w} \cdot \mathbf{x_i}) + b \geq 1 - \xi_i. \qquad (2.18)$$

By modifying the objective function we can simultaneously minimize the controller capacity while taking account of the extent of the constraint violation. The new function becomes

$$minimize\ \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \qquad (2.19)$$

subject to the constraints (2.17), (2.18) and $C > 0$. Here C dictates the penalty for the outliers. Introducing Lagrangian multipliers $\alpha_i \geq 0$ and

$\beta_i \geq 0$, the Lagrangian becomes.

$$\begin{aligned} L(\mathbf{w}, b, \xi, \alpha, \beta) &= \|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i}^{l} \alpha_i \left(y_i \cdot ((\mathbf{x_i} \cdot \mathbf{w}) + b) - 1 + \xi\right) - \\ & \sum_{i=1}^{N} \beta_i \xi_i. \end{aligned}$$

Differentiating with respect to the primal variable ξ,

$$\frac{\partial}{\partial \xi_k} L(\mathbf{w}, b, \xi, \alpha, \beta),$$

reveals in addition to (2.6) and (2.7)

$$C - \alpha_k - \beta_k = 0.$$

Since all $\beta_k \geq 0, C > 0$, all $0 \leq \alpha_k \leq C$. Further, if $\xi_k > 0$ then $\beta_k = 0$ (because the Lagrange function is to be maximized with respect to β). Also $\beta_k = 0 \Rightarrow \alpha_k = C$, so for a data point x_k with $\xi_k > 0$ the corresponding $\alpha_k = C$. The dual problem again becomes maximizing (2.14), but the constraints change slightly

$$0 \leq \alpha_i \leq C,\ i = 1, \ldots, l,\ and\ \sum_{i=1}^{l} \alpha_i y_i = 0.$$

The upper bound on α limits the influence of individual data points. The constant b can be calculated through (2.16) but one must choose a data point with $0 < \alpha_j < C$ i.e a support vector with $\xi_j = 0$.

This concludes the section on Support Vector Machines. Using SVM's thus includes selecting an appropriate kernel and calibrating the parameter C together with kernel parameters.

2.2 Basic mathematical tools for image analysis

In this section some basic tools for image analysis is described. For the complete theory see the respective references.

2.2.1 Image histogram

An image histogram is a graph over how frequent the different intensities in the image are. A histogram for a gray-scale image is a graph in two dimensions with the intensity on the x-axis and the frequency (number of pixels with that intensity) on the y-axis.

2.2.2 Thresholding

Thresholding is a basic segmentation technique, where a gray-scale image is transformed into a binary image. Let $g(x, y)$ be a gray-scale image and $b(x, y)$ a binary. The thresholding is then defined as

$$b(x,y) = \begin{cases} 0 & g(x,y) \leq T, \\ 1 & g(x,y) > T, \end{cases}$$

where T is the threshold value. The Otsu method described below is a way of choosing the threshold value dynamically.

2.2.3 Otsu's thresholding method

In [7] Otsu addressed the problem of finding a threshold value in a gray-level histogram without having any a-priori information about the image. The method uses a probabilistic approach where it selects the threshold value that gives the best separation between the two classes under certain criteria. For a complete theory see [7], or [8] for an excellent overview. The Otsu method is fast and can be proven optimal under certain normality assumptions [10]. It only works for problems with two classes but can be extended to multiple class problems at increased computational cost. In [9] a simple and fast technique for doing this is described.

2.2.4 Mathematical morphology

Mathematical morphology is a tool for extracting image components that are useful in the representation and description of region shape, such as boundaries, skeletons, and the convex hull. Concepts such as set theory, topology and geometry are used. In this thesis two basic morphological operations are used; opening and closing. Opening is a way to separate two objects that are slightly connected, removing thin protruding parts and smoothing the contour. Closing removes small holes, fills gaps and merges narrow breaks. For the theory see [11].

Chapter 3

Feature selection and SVM calibration

This chapter describes the work that was done when developing the method. It is divided into four parts; finding and evaluating features, creating a segmentation function, training/calibrating a support vector machine, and evaluating which features are more and which are less important.

3.1 Feature selection and evaluation

The features are divided into three parts, features that measure the contrast in the image, features that characterize the image content and a feature that determines if the image is negatively or positively defocused. When selecting the features the basic idea was to rather include too many than too few as the SVM, given enough data, in theory should be able to disregard those that do not provide relevant information. A complete list of the features used is provided in Appendix A.

3.1.1 Contrast features

There is a lot of material written on the subject of auto focus. Different algorithms work together with different metrics to find the best focused image. The part about algorithms are not relevant for this thesis but an evaluation on different focus functions is necessary to find which are best suited for the problem at hand. First, let's look at what properties are desirable for this kind of metrics. The following list is presented in [4].

C1 Accuracy- the maximum of the function must coincide with the best focused image.

C2 Unimodality - there should be no other (local) maxima than the global maxima.

C3 Reproducibility - the maxima should be as sharp as possible.

C4 Speed- the method should be fast enough to be useful in real time applications.

C5 Object texture- Robustness with respect to object texture.

There are many types of focus functions that satisfy the criteria above to a greater or lesser extent. In [6] three categories of focus algorithms are presented.

Focus functions based on differentiations:

$$F^1_{n,m,\Theta} = \int_{image} E\left((\left|\frac{\partial^n g(x,y)}{\partial x^n}\right| - \Theta) \right)^m dxdy, \tag{3.1}$$

in which $g(x,y)$ is the gray level at (x,y), Θ is a threshold and $E(z) = z, z \geq 0$; $E(z) = 0, z < 0$. These functions measure the amount of high frequencies in the image by differentiating the image and adding the differentiated values together.

Focus functions based on Depth of Peaks and Valleys:

$$F^2_{f,\Theta} = \int_{image} f[g(x,y) - \Theta] dxdy, \tag{3.2}$$

where $f(x)$ is chosen either as the threshold function or a similar function. This method works on the assumption that the gray levels in a focused image vary more than they do in a defocused. Note that no spatial information is used.

Focus functions based on the variance in the image

$$F^3_{m,c} = \frac{1}{c} \int_{image} |g(x,y) - \overline{g}|^m \; dxdy, \tag{3.3}$$

where $\overline{g}$ is the average gray level in the image. This method doesn't require any spatial information either but has the advantage that is doesn't use a threshold, which is hard to find automatically, so it is more robust to different image contents than (3.2).

From these three categories a multitude of functions can be constructed. Some of these are selected and tested, and those that work best are chosen as features.

From (3.1) two functions are implemented and evaluated. With parameters $\Theta = 0, m = 2, n = 1$ it's called the square gradient function. With $\Theta = 0, m = 1, n = 1$ it's called the absolute gradient function. These both have the advantage of not requiring a threshold value. By squaring the derivatives a sharper peak appears for the best focused image, catering to item **C1** in the list above. Therefore the square gradient function works better and is selected as a feature. The absolute gradient function is faster to calculate but this is considered less important.

From (3.2) two functions are also implemented and tested. One with $f(x) = E(x), \Theta = 0$, called Threshold video-signal content, and one with $f(x) = 1, x > 0, f(x) = 0, x \leq 0, \Theta = 0$, called Threshold video-signal pixel count. Additionally a method, called Entropy in [4], which has similar characteristics is implemented and tested

$$F_{entropy} = -\sum_{k} p_k log_2 p_k,$$

where p_k is the relative frequency of gray level k. These methods give unsatisfactory results with regards to the criteria listed above, and none of them are selected as features.

From (3.3) three different settings are tested. With $m = 2$ and $c = A =$ image area (3.3) becomes the standard variance. By setting $c = A \cdot \overline{g}^2$ a compensation for the average image brightness is achieved. It's then called the normalized variance. With $m = 1$ and $c = A =$ image area, (3.3) becomes the absolute variance. Variance and normalized variance work similarly while absolute variance is outperformed by the two others. These work well with regards to **C2** but not so well with regards to **C1** and **C3**. These are thus not included as features in this section, but the standard variance is included as feature in section 3.1.3 as part of characterizing the image content.

Apart from the above mentioned methods, three other focus metrics are evaluated. The first two are the Vollath's F_4 [4]

$$F_{voll4} = \sum_{i=1}^{M-1}\sum_{j=1}^{N} g(i,j) \cdot g(i+1,j) - \sum_{i=1}^{M-2}\sum_{j=1}^{N} g(i,j) \cdot g(i+2,j),$$

and Vollath's F_5 [4]

$$F_{voll5} = \sum_{i=1}^{M-1}\sum_{j=1}^{N} g(i,j) \cdot g(i+1,j) - \overline{g}^2 \sum_{i=1}^{M}\sum_{j=1}^{N} 1.$$

Vollath's F_4 is based on the autocorrelation function and Vollath's F_5 is based on the standard deviation function. The first function lives up to

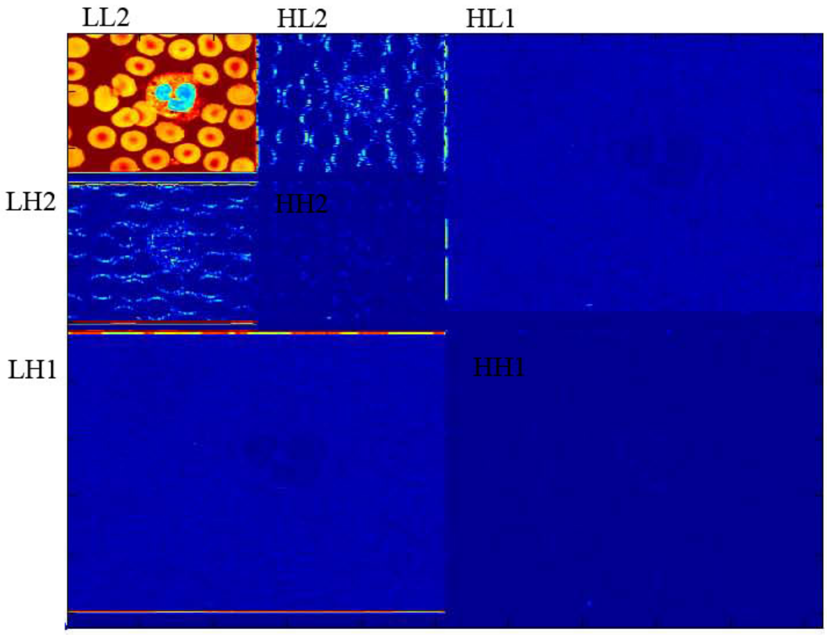

Figure 3.1: The result of a two dimensional two-level wavelet transform

criteria **C1** quite nicely but have some problems with **C2**. The second works the opposite way. Because criteria **C1** is regarded as the most important one Vollat's F_4 is selected as a feature. Also, a self composed mix of the F_4 and F_5 that combines the advantages of both models, is used.

$$F_{mix} = \sum_{i=1}^{M-1}\sum_{j=1}^{N} g(i,j)\cdot g(i+1,j) - \frac{19}{20}\sum_{i=1}^{M-2}\sum_{j=1}^{N} g(i,j)\cdot g(i+2,j) - \frac{1}{20}\overline{g}^{2}\sum_{i=1}^{M}\sum_{j=1}^{N} 1.$$

The third metric is based on wavelets. This focus function is based on ideas described in [1] and [2]. The theory behind wavelets is not provided here, see chapter 7 in [11] for an introduction. The wavelet based focus metric developed in this thesis works as follows. First the size of the image is changed from original 480 by 640 pixels to 512 by 512 through cutting away columns and adding empty rows. The value in each pixel is set to the mean of the r,b and g color layer, creating a gray level image. Then a two dimensional two-level wavelet transform with a Daubechies 8 mother wavelet is performed through using the Wavelab 850 toolbox for Matlab developed by the WaveLab Development Team at Stanford University. The result from running this file is presented in Fig. 3.1.

What has happened is basically this; first the images is subsampled to a quarter of its original size. HL1, LH2 and HH1 contain the differences between this first subsample and the original image. Then the subsampled image is subsampled again to a 1/16 of the original size. HL2, LH2, and HH2 contain the differences between the double subsampled image (shown in LL2) and the first subsampled image. Only parts HL2 and LL2 will be used in the rest of the method. The idea is that the sharpness of the edges in HL2 is a measure of how sharp the original image is. Hence; the 280 pixels with the highest intensity in HL2 are found, and the twenty with the highest are thrown away to reduce noise impact. The intensities of the remaining 260 pixels are added together and divided by the mean of the image LL2. The division is done for normalizing purposes. As explained earlier a bright image with many details will have sharper edges than an equally sharp paler image with few details.

The method works very well with respect to **C1** and **C3** but it has problems determining which is the more defocused between two heavily defocused images.

Also, two focus functions developed earlier at Cellavision, Focusvalue and Blurvalue were used. Focusvalue gives reliable and sharp peaks while blurvalue has the characteristics of an absolute focus metric giving roughly the same value for focused images with different image content. In Fig. 3.2 the six chosen focus functions are applied to the images series that contain the images presented in Fig. 1.3 on page 6. The red lines with + signs contain image A, the blue with dots contain image B, the green lines with rings contain image C, and the black lines with crosses contain image D. The peaks are generally correctly positioned on the best focused image (with $z - f = 0$). All functions except the wavelet based function have some difficulties with the series containing image A due to the low level of details in these images. The green color layer is used in all cases except the wavelet based where a mean of all color layers is used.

Because the WBC is higher and bigger as was shown in section 1.4, the image (in a series of images taken at different distances z) with the highest global contrast doesn't necessarily have the highest contrast at the WBC. While the difference is rather small, it is still important as the WBC is the important part of the image. Also, given both the contrast in the WBC and the global contrast, one can draw conclusions on the direction of defocus according to Section 1.4. To address this, four of the features selected in this section are also applied to an rectangle around the largest WBC in the image. The Wavelet based measure and the Blurvalue are not used as they need a larger area to work properly. The area of this rectangle is also included as a feature for the purpose of normalizing the measurements.

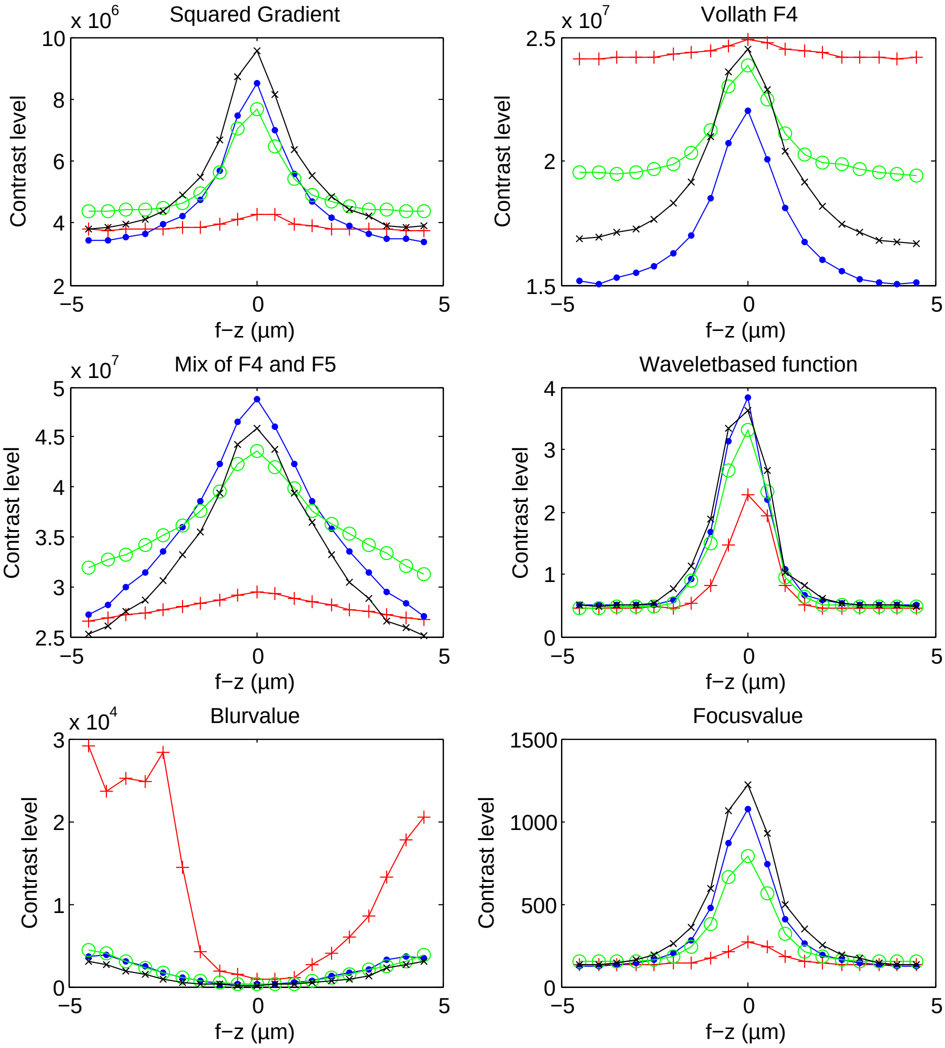

Figure 3.2: The six focus functions chosen to measure contrast level. All applied to the green layer of the images in Fig. 1.3. The red lines with + signs contain image A, the blue with dots is contain image B, the green lines with rings contain image C, and the black line with x'es contain image D. The images are shown in Fig. 1.3 on page 6

3.1.2 Directional features

As was described in section 1.4 the different color layers in the image are focused at different distances z. This can be used to determine the direction of defocus. As noted in section 3.1.3 the variance is already calculated for the different color layers as part of the content features. This could very well give the SVM enough input to tell direction of defocus. However, a method, DirValue, already developed at Cellavision and used in the current system as a hint to the direction of defocus, is included as a feature, even though it doesn't work reliably by itself. This is because it may work more robustly when combined with information from the other features.

3.1.3 Content features

The following features try to characterize the content of the image, giving the SVM a chance to normalize the measurements in the previous sections. These can be divided into two groups, those that depend on a segmentation of the image, and those that work on the whole image. Together with a team from Cellavision, a list of possible features was created.

- Mean intensity for the whole image
- Mean intensity for the background
- Mean intensity for the RBCs
- Mean intensity for the WBCs
- Images standard deviation, STD, for the whole image, calculated as

$$STD = \sqrt{\sum_{allpixels} (pixelvalue - meanpixelvalue)^2}.$$

- STD for the background
- STD for the RBCs
- STD for the WBCs
- Number of pixels occupied by RBCs
- Number of pixels occupied by WBCs

All brightness and variation measures are calculated for all three color layers. Initially the number of RBCs and the number of WBCs were also included in the list, but these features proved to be too difficult to implement robustly so they were taken out. The problem was quite simply that with a great level of defocus two adjacent RBCs, or WBCs, melt together and form one.

Separating these in a robust way is quite a hard problem to solve and was neglected due to time limitations, and due to the fact that satisfactory results were reached without using these features. The image standard deviations could very well be regarded as contrast features, but they are inferior as such to the ones selected in that section. However, they are important when characterizing the image content and that is why they are included in this section.

3.2 Image segmentation

Several of the items in section 3.1.3 depend on that the computer knows which pixels represent RBCs, WBCs and background. Before they can be calculated an image segmentation has to be done. The ambition is to create three binary masks for pixels containing the background, the RBCs and the WBCs respectively. The cytoplasm, which is the part of the WBC that is not the core, is not to be included in these masks and neither is dirt nor color stains. The method has to be robust enough to handle heavily defocused images. In the rest of this section, WBC will refer to the core of the WBC.

First, let's look at some basic characteristics of a typical image. In Fig. 3.3 the histograms for the images in Fig. 1.3 are shown. On the x-axis is the intensity of the green color layer and on the y-axis the number of pixels with intensity x. The big peak to the right represents the background, which is the lightest object. The smaller peak next to it is from the RBCs. In sub image B this peak has two peaks due to the intensity difference between the center and the perimeter of the RBCs. The small lump around $x = 40$ is the WBC. The pixels that lie between the RBC peak and the WBC peak is either the cytoplasm, dirt, or color stains (as in sub image C). The cytoplasm has very similar colors to the RBC and it is sometimes hard, or even impossible using no spatial information (e.g. closeness to the core), to segment the two from each other.

Next, let's make an analysis of how the colors vary in the different parts of the image. In Fig. 3.4 B is a plot of how the different color layers vary along the line in Fig. 3.4 A. The red solid line represents the red color layer, the green dash-dot the green color layer and the blur dotted line the blue color layer. Notice how the blue color layer varies less than the others and also that the green color layer has a lower intensity than the blue only for pixels in the WBC. Sub image C shows how the intensity of the green color layer minus the intensity of the blue color layer varies along the line in sub image A.

This opens up for segmenting the WBC in a fast and simple way; through subtracting the blue color layer from the green and then threshold at zero.

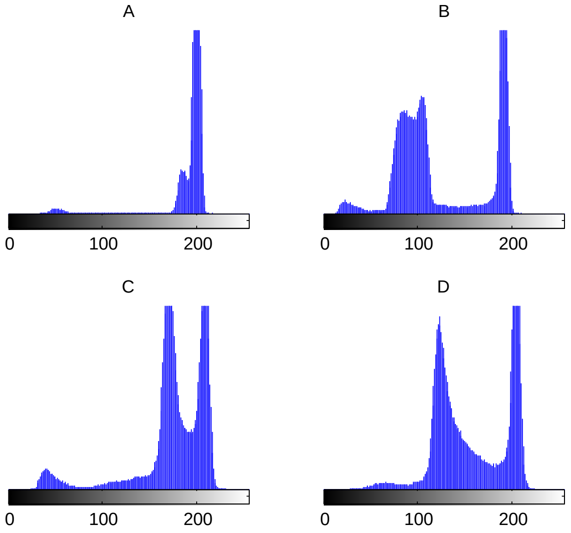

Figure 3.3: Histogram for the images in Fig. 1.3

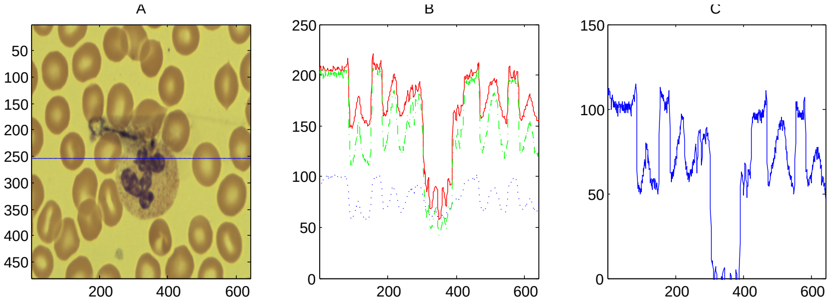

Figure 3.4: Analysis of how the color components vary in different parts of the image. The red solid line represents the red color layer, the green dash-dot the green color layer and the blue dotted line the blue color layer.

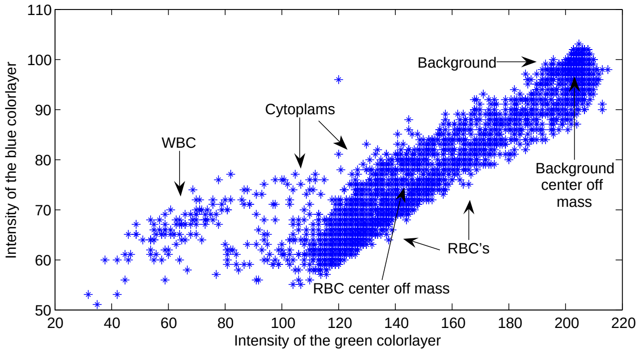

Figure 3.5: Plot of the intensity of the blue color layer versus the green.

As it turns out, this doesn't work for all types of WBC and a more advanced approach has to be used.

After analyzing plots of the RBC color space the following approach was used. If one plots the intensity of the blue color layer against the green one will get the result illustrated in Fig. 3.5. Maximum separation of the WBC from the rest of the image is achieved if the pixels are projected from the blue green color space onto a vector perpendicular to the direction from the RBCs center of mass to the background's center of mass.

With these three findings a segmentation algorithm was composed that works robustly, but not perfectly, for all images in the data. This algorithm works in the following steps.

First a rough estimate of the WBC is made by thresholding the difference between the green and blue color layer at zero. This result is then smoothed and isolated pixels removed.

Next the WBC-pixels are removed from the image and the background is separated from what remains through thresholding the green color layer at a threshold value found using Otsu's method. Now remains only to separate the RBCs from trash and other oddities. This is done by finding the most frequent intensity (this will be in the middle of the RBC peak), and then using all pixels that have intensities that are higher or slightly lower. This way trash that has a lower intensity is filtered away. The cytoplasm, however, will in many cases still be included as it often has the same intensity as the RBCs.

After these initial estimates the more thorough approach described above is used to pinpoint better which pixels describe the WBC. At this stage the WBC estimate is reasonably accurate, but in some cases part of the cytoplasm is still included. The pixels describing the WBC are therefore separated again using a threshold again found with Otsu's method. The result is then smoothed and noise is removed. At this stage the WBC estimate is quite accurate and to reduce cytoplasm influence on the RBC mask all pixels in a surrounding of the WBC are said to be non RBC pixels. Some pixels that describe RBCs are thus wrongfully removed but, more importantly, large parts of the RBC mask that actually was cytoplasm is removed. Finally, also the RBC mask is smoothed and isolated pixels are removed.

3.3 Calibrating and training the Support Vector Machine

After selecting and extracting features the work on calibrating and training the SVM began. A complete system was constructed that loaded the images, made the segmentation, extracted features, and then trained a SVM on one part of the material and tested it on the other part. This way an improvement in one part of the algorithm could be easily evaluated. Initially, for example, there were several problems with creating a robust segmentation algorithm and this was instantly reflected in the success rate of the SVM to classify the unknown images correctly.

The database that initially was trained and tested on, Database1, consists of 4902 images in 248 series. Each series consist of 19 images of the same object, with distances to focus

$$(z - f) = [4.5, 4, \ldots, 0.5, 0, -0.5, \ldots, -4, -4.5]\mu m.$$

Series were selected that contain as many of the variations as possible according to the list in section 1.4. The only restriction is that the series used all contain one or more WBC. This was done to make it easier to create a robust segmentation algorithm, see section 3.2 on page 26, and is something that has to be addressed should the method be used in real applications. The images were captured in the following way: Several images of the same object were captured at various distances, z. Then the contrast was measured using "focusvalue", see Section 3.1.1. Doing this the outer perimeter of the image was disregarded to emphasize the part of the image that contains the WBC. The image with the highest contrast was selected as focused and images at distances $(z - f) = [4.5, 4, \ldots, 0.5, 0, -0.5, \ldots, -4, -4.5]\mu m$ were saved with a label showing how defocused they were. All images have 480 by 640 pixels.

When training and calibrating a SVM one always needs to keep in mind if a misclassification in one direction is worse than one in the other direction.

In this case it is considered worse if the auto focus algorithm classifies a defocused image as focused than the other way around. This is because the next step in the system, the WBC-classifier, will not function correctly if it receives a poorly focused image. If the focus metric, on the other hand, does not recognize a focused image as focused, the process is still in the hands of the auto focus routine and can be dealt with.

After having gone through all images and extracted the features, the next step was to normalize the data. This is done to increase the numerical stability of the method, and to prevent features with large numerical values to dominate over those with small. The data used for training was linearly scaled between 0 and 1. The data used for testing is then scaled using the same method. For example, let's say that one feature in the training data was scaled from [10, 20] to [0, 1]. If the same feature in the test data lies between [11, 19] it will be scaled to [0.1 , 0.9]. After this the calibration of the SVM could begin. Two different kernels types were evaluated, the radial basis function kernel, K_{RBF}

$$K_{RBF}(\mathbf{x_i}, \mathbf{x_j}) = e^{-\gamma|\mathbf{x_i}-\mathbf{x_j}|^2} \tag{3.4}$$

and polynomial kernels, K_{poly}

$$K_{poly}(\mathbf{x_i}, \mathbf{x_j}) = \gamma(\mathbf{x_i} \cdot \mathbf{x_j})^d \tag{3.5}$$

of dimension $d = 1 \ldots 7$. For each kernel there are two variables that can be adjusted, the C variable from equation (2.19), and γ from equations (3.4) and (3.5). Also, it is possible to put weights on C to penalize outliers from one class higher than outliers from another. The testing was done so that for a certain kernel and certain choice of weights, different combinations of C and γ were tried in a logarithmical gridsearch. Thus all combinations of $C = 2^{a,a+1,\ldots,a+n}$ and $\gamma = 2^{b,b+1,\ldots,b+m}$ were tried where, $a, b, n\ and\ m$ were chosen to well cover the values that generated the best results. For all these tests a cross validation method was used. The data was divided into five parts where four were used to train on and the fifth to test on. This was repeated five times and the mean result was used to get a measure on how good a certain choice of parameters and kernel was. Also, before the data was divided into these five parts, it was rearranged randomly so that data from all slides was in each of these five parts. The reason for this is illustrated with an example: If there were a total of 20 series with exceptionally low contrast in the whole database, and these all went into the same fifth, the SVM would have a hard time classifying these correctly as there would be no training examples in the other four fifths.

Carrying out these tests a way to compare results was needed. It was decided that for the two class problem a defocused image being classified as focused is ten times worse than the other way around. For the multi class problem the following was decided:

- Being one step wrong is penalized with one point. Being two or more with two points.
- If a defocused image is classified as focused the penalty is multiplied by ten.
- If a focused image is classified as defocused the penalty is multiplied by five.

For example: a image with $(z-f) = 0.1\mu m$ that is classified as focused gets twenty point penalty while a image with $(z-f) = 0.2\mu m$ that is classified as $(z-f) = 0.15\mu m$ only gets one point penalty.

The construction of these rules was based on the idea that a focused image should be captured on the second try. Therefore it doesn't matter if the image is classified two or four steps wrong. It is better, though, if it classifies only one step wrong as the next image might be focused anyway due to vibrations and uncertainties in the system. The second item is already explained and violation is heavily penalized. The third item was added because if a focused image is falsely classified, the system will make a detour and can earliest find focus in three steps.

After testing a number of images each misclassified image is penalized as described above, the penalties are added together and then divided with the number of images in that test run. This way a scalar measurement, which will be referred to as the error value, is created that indicates how good the method works. An error value of 1 means that, in average, all images were misclassified one step.

3.4 Reducing the number of features

After finding out which kernel and which choice of parameters works best a reduction of the number of features began. The reason for doing this is that some features probably don't provide any useful information and removing these can improve the results. Also, the speed of the process is improved if fewer features have to be extracted and put into the SVM. In particular, if all segmentation depending features could be removed, the speed and robustness of the method could be improved significantly. When doing the features reduction Database1 was used.

Preferably, one would like to try all combinations of all 65 features. This would mean testing

$$2^{65} = \sum_{k=0}^{65} \frac{65!}{k!\,(65-k)!} \approx 4 * 10^{19}$$

combinations. Testing one combination requires 3 seconds on the PC described in Section 1.1, and needs to be done for the five divisions of data and for at least 5 by 5 choices of C and γ around those values that are optimal when using all features (the last is done because the optimal choice of parameters shifts when using fewer features). Doing this would take $5 * 10^{14}$ years and instead an iterative approach has to be used.

First, three tests were done. A reduction analysis, an addition analysis and a sensitivity analysis.

The reduction analysis was done as follows. Beginning with all features, one by one was temporarily removed and the error value was calculated using the other 64 features. The feature that, when it was removed, allowed the remaining features to generate the smallest error value, was regarded as the least valuable feature and was permanently removed. The remaining 64 features were then processed in the same way removing one feature at a time until only one remained. This process required evaluation of roughly $65 * (65 + 1)/2 = 2145$ combinations and was done in ≈ 9 days.

The addition analysis worked in the opposite way. Beginning with only one feature, the one that generated the smallest error value when used by itself was kept. Then the feature that, together with the ones that were already chosen, generated the smallest error value, was added. This was repeated until all 65 features had been added one by one. This process requires the same number of combination as the reduction.

The sensitivity analysis evaluated the features by measuring how sensitive the error value is to variations in that feature. First, the training was done as usual using all features. Then, one at a time, the features in the test data were increased with one standard deviation of that respective feature. The error value was then calculated with the modified features, and this was repeated for all 65 features. If, for example, the error value isn't affected when a feature is changed, this feature is probably not very important to the performance of the system. It might, however, be the case that two features provide similar but crucial information and while changing one doesn't have a great impact changing both would have. These kinds of connections between features are not found with any of the methods used and no account could be taken of them.

In Fig. 3.6 are plots of the addition and reduction analyses. It is obvious that all 65 features do not seem to be necessary. The minimum error value for the reduction analysis is 0.1703 when using 38 features, and the minimum error value for the addition analysis is 0.1779 when using 46 features. In Fig. 3.7 is a plot of the sensitivity analysis. Drawing conclusions from this material is quite difficult. One thing is that the optimal number of features seems to be be between 38 and 46. Also, if one compares the features sets

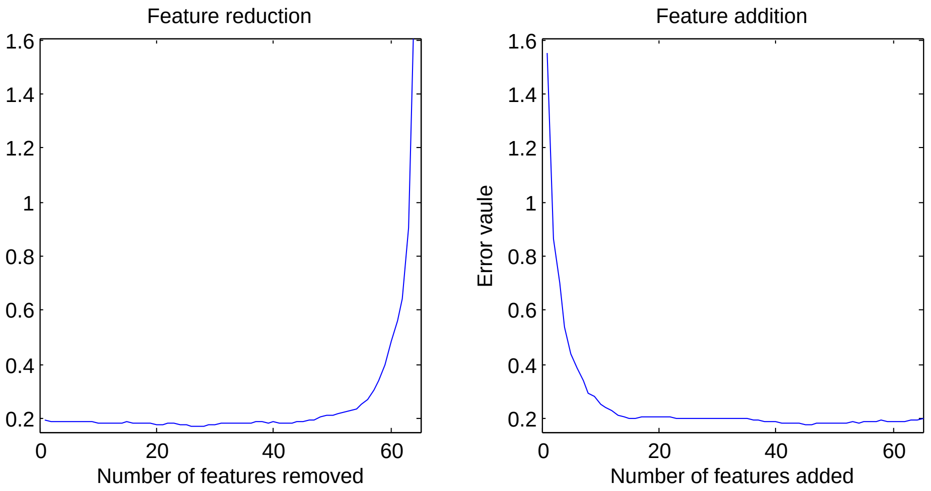

Figure 3.6: Plots of how the error value varies with the number of features

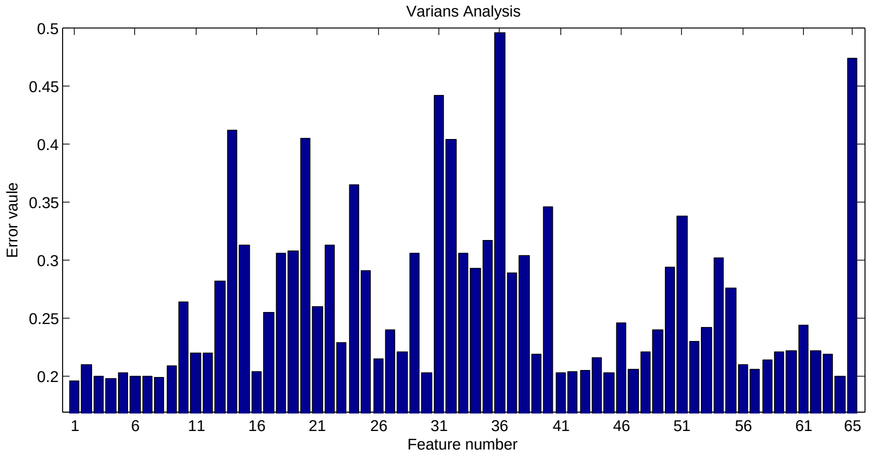

Figure 3.7: Plot of a how great impact a feature variation has on the result

from the reduction and addition analyses that generated the best results two things become evident.

First, the number of features from each color layer is almost the same in both cases, and also evenly distributed between the different color layers. There are approximately 11 features from the red, green, and blue color layers and around 10 features that are based on the mean color layer or binary information.

Secondly, the kind of features used is not the same. The addition scheme prefers more content features, especially mean intensities while the reduction prefers contrast features. There is a particular big difference in the WBC contrast section with 3 features in the addition scheme and 11 in the reduction. Se appendix C a for a complete table of which category and color layer the selected features belong to for the respective analyses. Also, a list of 38 features that was generated with a random approach, which is described below, is presented.

These differences indicate that the content features can be used in small numbers to get a rough result, while contrast features are needed to reach the top performance. Also, given enough contrast features, they contain in them much of the information given by the content features.

Next, I try to use the results from the sensitivity analysis to further understand which features are important. Let's call the y-values in Fig. 3.7 sensitivity numbers, SN's. Two sets of 20 features are selected.

Set A is selected in the following way: Most features can be put into groups where the same calculation is applied to the three color layers, or even four, including the mean of all color layers. There are 17 such groups containing a total of 60 features. The other 5 features do not fall into one of these groups as they are applied either only to a mix of color layers or to a binary image. The features from each of these groups with the highest SN is selected for set A. Also features "Wavelet", "mangefokus" and "area RBC" are selected from the remaining five features as they have significantly higher SN's than the other two. Set B contains the twenty features that have the highest global SN.

These sets will now be compared with two other sets. Set C that contain the 20 best features from the reduction analysis and set D that contain the twenty best features from the addition analysis. Now let three new sets, sets E, F and G be defined as

$$\begin{aligned} E &= A \cap C \cap D, \\ F &= B \cap C \cap D, \\ G &= (A \cup B) \cap (C \cup D). \end{aligned}$$

Feature set	**Mean error value**	**STD**
Random	0.24	0.026
Set E	0.22	0.016
Set F	0.22	0.018
Set G	0.20	0.014

Figure 3.8: Results from the random feature analysis approach

E thus contains 5, F contains 7, and G contains 20 features. The features selected this way are assumed to be more important than the others, and to verify this a fourth test is done. The features in sets A to G are listed in appendix D.

It is assumed that 38 is a good number of features as this is the number for which the best result is achieved so far (in the reduction analysis). The idea is to randomly select 38 features from the original 65, calculate the error value from this selection and then repeated 1500 times. This test will then be run three more times, each time fixating the features from set E,F, and G and then adding randomly selected features (from the remaining) to get a total of 38 each time. If the sets contain well chosen features, the mean error value for these runs will be lower than when randomly selecting all features. To speed things up the cross validation method was not used but the SVM was only once trained on four fifths of the data and tested on the last one fifth.

The results of this test are presented in Fig. 3.8. In the first row all 38 features are randomly selected and in the following the features included in the mentioned sets are fixated. Using the features in Set G together with 18 randomly selected is thus in average 17 % better than randomly selecting all features. Considering that only 1500 combinations have been tried, and that the number of possible combinations are in the vicinity of 10^{12} to 10^{18} it is quite remarkable that the standard deviation is so low. This indicates that many features are heavily correlated and thus provide similar information.

To generate a final list of features the same kind of test was run again 500 times for Set G using the more thorough cross validation method. The list generated this way that gave the best result had an error value of 0.1826 which is good, but not better than the best results from the reduction and addition analyses. This list is also put into the same kind of table as the best feature sets from the addition and reduction analysis in appendix C. Notice how also in this case the features are evenly spread out among the three color layers.

The best results from the tests done in this section was thus achieved when using the 38 features that generated the lowest error value in the reduction analysis. This set will be used in the next chapter and will be referred to as the "reduced feature set".

This concludes the features selection section. Not enough has been done to be able to say exactly which nor how many features are needed for an optimal solution, or even with certainty which kind of features or feature groups are important. The items in the list below are therefore not facts but rather assumptions based on what I have observed.

- An even distribution from the different color layers seems to be preferred.
- The optimal number of features is less than 65 and larger than 20.
- The features in Sets E, F and G contain important information.

There is a lot more that can be done here but was neglected due to time restrictions. One thing is to make a correlation analysis between the features and thus reducing them more methodically than what has been done here. Another thing is to redo the reduction and addition analysis while fixating the features in sets E,F and G.

Chapter 4

Validation and Results

In this chapter the method developed in the previous chapter will be tested on different databases. The results are presented in Fig. 4.1 and more detailed in Figures 4.2 to 4.10. The specifications of the databases used are provided in the text but also repeated in appendix B.

4.1 Test preliminaries

The two problems formulated in the problem description were initially treated separately, but as the work proceeded two things became evident: The multi class problem is possible to solve with such satisfactory results that this method will be implemented in the system used at Cellavision AB. Also, when solving the multi class problem one effectively solves the two class problem as well, and calibrating parameters for best performance of the multi class problem gives satisfactory results (however not optimal) also for the other. Therefore, the focus when testing was on solving the multi class problem.

After testing the different kernels, weights, and parameters according to Section 3.3 it was concluded that a RBF kernel with $C \approx 2^8$ and $\gamma \approx 2^{-2}$ gave the best results (closely followed by a 2 dimensional polynomial kernel). The outliers in the different classes were penalized equally, i.e. no weights were used. Depending on the choice of features, normalization method and which database was used the optimal values for C and γ varied. These parameters were calibrated for each test so that $C = 2^n, \gamma = 2^m$ gave the best results for $n \in N, m \in N$, but not for $n \in \Re, m \in \Re$ which most likely would improve the results further.

4.2 Method validation

Up to here, only Database1 has been used. After achieving satisfactory results working only with this material additional testing was done. If this system is to be useful in real applications a common training data has to work for test data from different laboratories. The training data will have to vary according to the list in Section 1.4 while the test data will vary to a smaller extent. This is because a certain lab always uses the same smear and stain methods. The variations in a certain test data will thus be a subset of the variations in the training data.

To properly evaluate the method testing data was collected from a single laboratory. This data will be referred to as Database2. Database2 consists of 12825 images in 513 series collected with the DM96. Each series consists of 25 images, with distances to focus

$$(z - f) = [6, 5.5, \ldots, 0.5, 0, -0.5, \ldots, -5.5, -6]\mu m.$$

Data with greater level of defocus than in Database1 was included to see how the classifier reacts to this. The best case is that the heavily positively defocused images all classifies in the category $(z - f) = 4.5\mu m$, and the heavily negatively in the $(z - f) = -4.5\mu m$. The second best is that they are all classified as $|z - f| = 4.5\mu m$ which then could be used in a control algorithm as ”do not trust the classifier if images fall into this group”. The worst case is if the heavily defocused images classifies in all groups, in which case the method basically will be useless.

As it turned out, the results got significantly worse when the SVM was trained on the Database1 and tested on Database2. This was believed to be because the Database1 is too small and because it was collected with the DM8 (as oppose to Database2 which was collected with the DM96). Therefore a third database, Database3 was collected. This database is collected with the DM96 and consists of 13129 images in 691 series. Each series consists of 19 images, with distance to focus

$$(z - f) = [4.5, 4, \ldots, 0.5, 0, -0.5, \ldots, -4, -4.5]\mu m.$$

The images in Database3 also vary, just like Database1, according to the list in Section 1.4 on page 4.

4.3 Results

In the table below the error values for 12 different tests are presented. In the first row Database1 has been used both for training and testing according to

Database Train/Test	**All features**	**Reduced features**	**Seg. independent features**
Database1/Database1	0.197	0.170	0.261
Database1/Database2	1.163	1.094	1.327
Database3/Database2	0.276	0.275	0.597
Database3/Database3	0.193	0.198	0.226

Figure 4.1: Error values for the multi class problem using different Databases and different choices of features

the crossvalidation principle described in Section 3.3. In the second row the SVM was trained on Database1 and tested on Database2. In the third row it was trained on Database3 and tested on Database2. In the fourth, it was trained on three fourths of Database3 and tested on the last fourth. In the first column all features are used. In the second column the reduced feature set from Section 3.4 is used and in the third only segmentation independent features are used. For a complete analysis of which method to use a similar table should also be made with the execution times. This has not been done in this thesis, but is something that needs to be done to be able to match method accuracy versus method speed properly. Here is only noted that the results in column two are a little bit faster to calculate than those in column one, and that those in column three are much faster to calculate than the two others. This is because when calculating the results in column three the image does not have to be segmented.

The following are important observations from Fig. 4.1.

- When using the reduced feature set the results improved by 14% in relation to row1.
- However, the reduced feature set (that was extracted using Database1) does not improve results when used on Database3.
- When testing on Database2, training the SVM on Database3 gives much better results than when training it on Database1.
- Using only segmentation independent features causes a $20 - 30\%$ decrease in method performance.

These observations will be discussed in the following sections where the results are looked at more closely, one row at a time, beginning from the top.

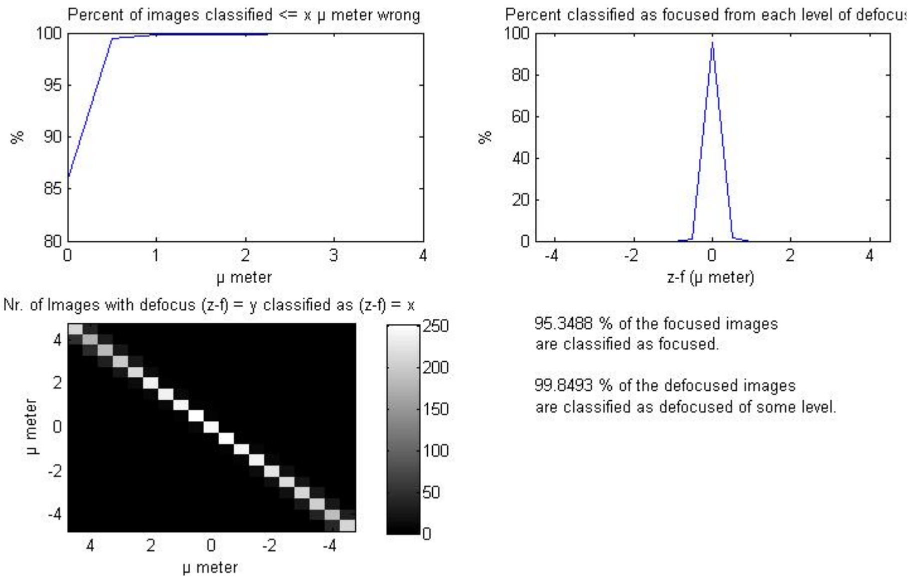

Figure 4.2: Results from Training/Testing on Database1 using the reduced feature set

4.3.1 Database1/Database1

The most important conclusion from this test is that there is a potential for improvement of the method by selecting which features to use. The error value decreased with 14% from column one to two, and like stated in Section 3.4, there is more to be done with regards to selecting which and how many features to use. This might lead to an even lower error value. When using features that do not depend on the segmentation, the result got 32% percent worse compared to column one. This appears like a severe performance decrease, but the results might be considered good enough to use anyway. The best result in row 1 was thus received when using the reduced feature set. To understand this result better, a more detailed result is illustrated in Fig. 4.2. The results are satisfying in many ways. First of all, only 0.15% of the defocused images are classified as focused, and these are only from the smallest level of defocus. Second, a much as 95.3% of the focused images are classified as focused. All together, 99, 4% of the images are classified less or equal to $0.5\mu m$ wrong.

What might not be obvious in Fig. 4.2 is that the precision is much better for the less defocused images. To illustrate this, the plots in Fig. 4.3 were made from a subset of the data used in Fig. 4.2. Only images with $|z-f| \leq 2.5\mu m$ were used. Note how almost 95% of the images are classified correctly and how virtually no images are misclassified more than $0.5\mu m$.

Finally, for the results from the third column in row 1 cf. Fig. 4.4. The

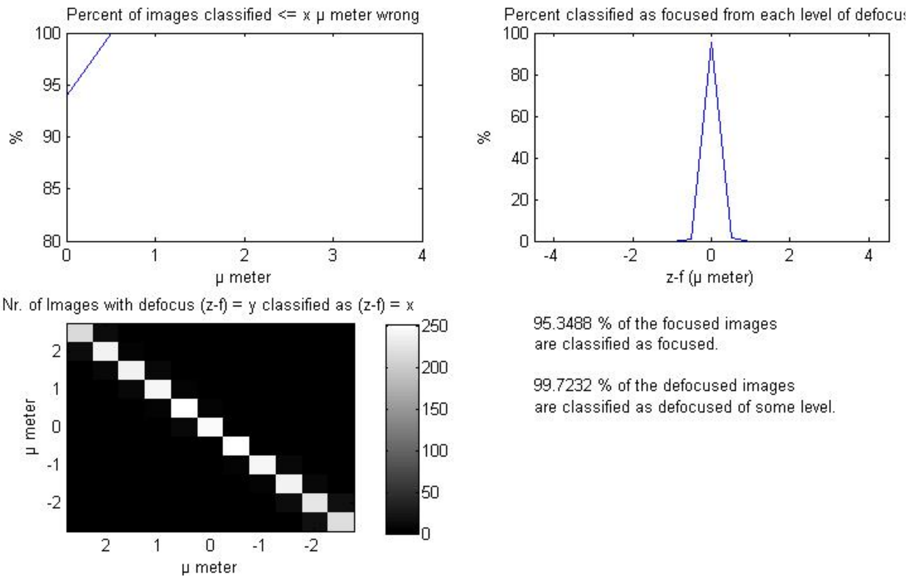

Figure 4.3: Results from Training/Testing on images with $|z - f| \leq 2.5\mu m$ from Database1 using the reduced feature set. Note how over 90% of the images are classified correctly and how virtually no images are misclassified more than $0.5\mu m$.

accuracy is not as good as in Fig. 4.2 but the decrease could perhaps be accepted due to the increase in speed this allows for.

4.3.2 Database1/Database2

When training on Database1 and testing on Database2 two things were to be evaluated. How does the classifier react when it is given more defocused images than it has been trained on, and how well can it classify the images that have the same degrees of defocus as the images in the training set.

The testing showed that the heavily defocused images with $(z - f) = [6, 5.5, 5, -5, -5.5, -6]\mu m$ were all classified as $(z-f) = 4.5\mu m$. The heavily negative defocused images were thus classified as positively defocused. This was not the best case scenario, but not the worst either, according to the criteria put forward in Section 4.2.

Looking only at images with

$$(z - f) = [4.5, 4, \dots, 0.5, 0, -0.5, \dots, -4, -4.5]\mu m$$

the error value using all features was 1.163, which is a significant decrease from the results in the previous subsection. Also here using the reduced feature improves the results somewhat, but not as much as in the previous

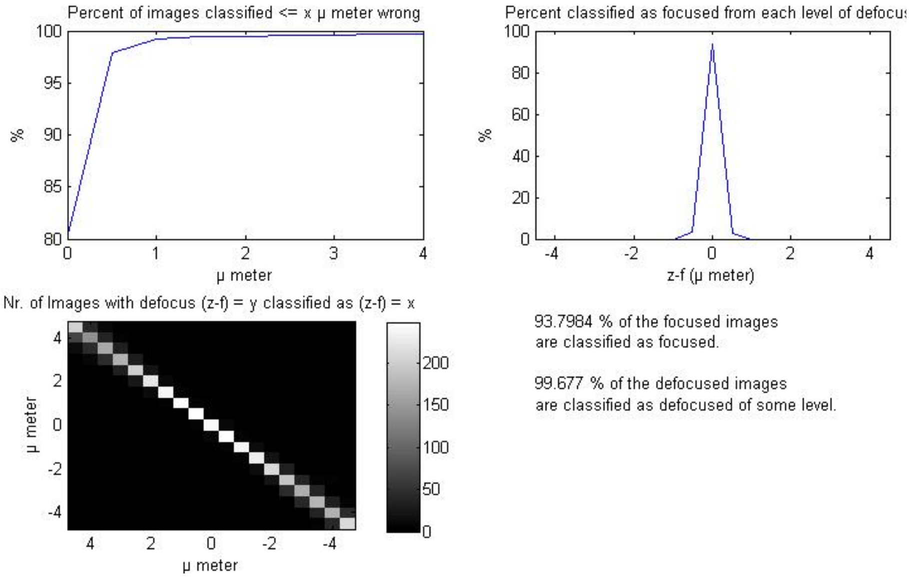

Figure 4.4: Results from Training/Testing on Database1 using only the segmentation independent features. The accuracy is not as good as in Fig. 4.2 but the decrease could perhaps be accepted due to the increase in speed this allows for.

section. Using only the segmentation independent features, the results got 14% worse. In Fig. 4.5 are the result details provided when using the reduced feature set. Note the significant decrease in overall performance compared to Fig. 4.2

4.3.3 Database3/Database2

When training on Database3 and testing on Database2 the results improved significantly compared to the previous section. This time the majority of the heavily defocused images with $(z - f) = [6, 5.5, 5]\mu m$ were classified as $(z - f) = 4.5\mu m$, and the ones with $(z - f) = [-5, -5.5, -6]\mu m$ as $(z - f) = -4.5\mu m$. This opens up for the possibility to train the method to distinguish between even more defocused images.

Looking only at images with

$$(z - f) = [4.5, 4, \ldots, 0.5, 0, -0.5, \ldots, -4, -4.5]\mu m$$

the error value was 0.275 which is a vast improvement from Section 4.3.2. Also this time the best performance was achieved using the reduced feature set, but the difference is neglectable. Using the segmentation independent features resulted in a 62% performance decrease. In Fig. 4.6 are the result details provided when using the reduced feature set. Note the overall

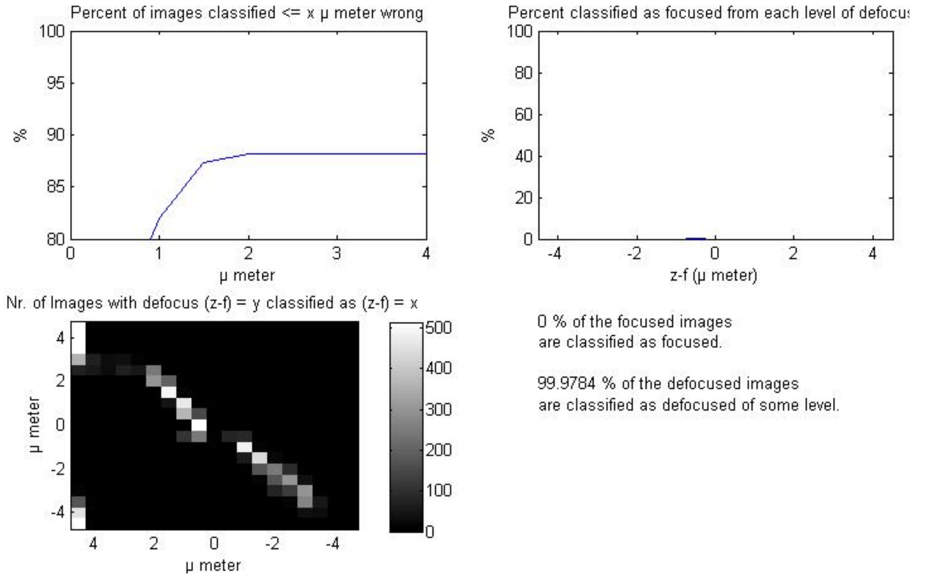

Figure 4.5: Results from Training/Testing on Database1/Database2 using the reduced feature set. Note the significant decrease in overall performance compared to Fig. 4.2

performance improvement compared to Fig. 4.5. Also note how ≈ 7% of the positively defocused images with $(z - f) = 0.5\mu m$ are classified as focused. The overall performance increase can most likely be explained by two things; that Database3 is bigger than Database1, and that Database2 and Database3 are collected using the DM96 while Database1 is collected with the DM8.

The misclassified images (with $(z - f) = 0.5\mu m$) should be investigated manually as images from this category sometimes are quite well focused on the WBC, and then good enough to use. Surprisingly many of the focused images (99%) are correctly classified.

Finally, for the results from the third column in row 3 cf. Fig. 4.7. There is a significant decrease in accuracy compared to Fig. 4.6, but again, this could perhaps be accepted due to the increase in speed.

4.3.4 Database3/Database3

This is the results of the final series of tests that was done in this thesis. It was done so that the SVM was trained on 3/4 of the data in Database3 and tested on the final fourth. The results from these tests, when using all features, are very similar to those in Section 4.3.1 and this is not so strange due to the similarity of the data used. Still, it was good to see that

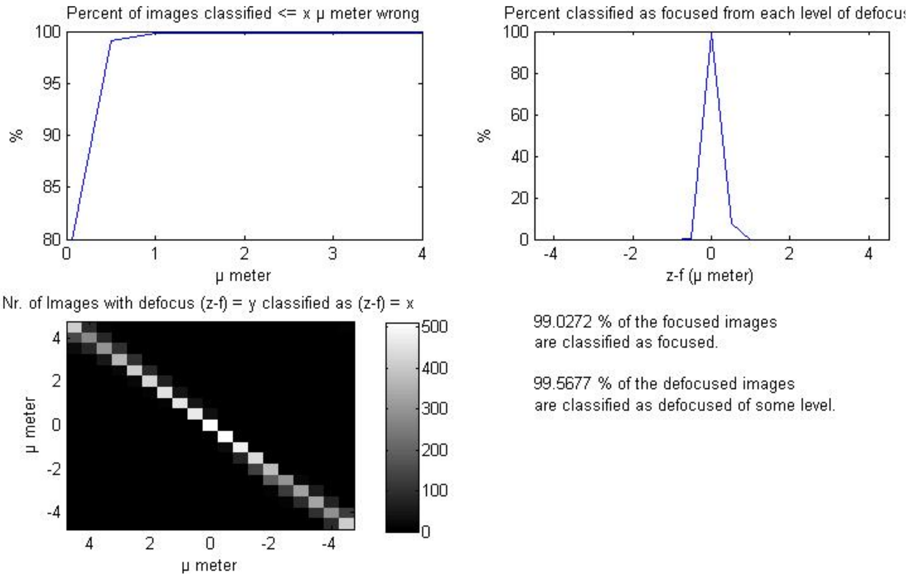

Figure 4.6: Results from Training/Testing on Database3/Database2 using the reduced feature set. Note the overall performance improvement compared to Fig. 4.5. Also note how $\approx 7\%$ of the positively defocused images with $(z-f) = 0.5\mu m$ are classified as focused.

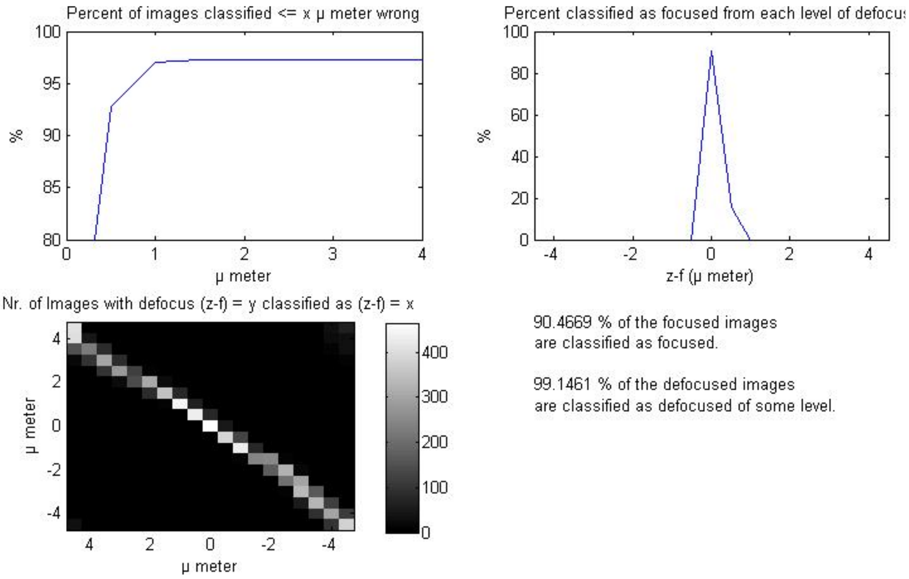

Figure 4.7: Results from Training/Testing on Database3/Database2 using only the segmentation independent features. There is a significant decrease in accuracy compared to Fig. 4.6, but again, this could perhaps be accepted due to the increase in speed.

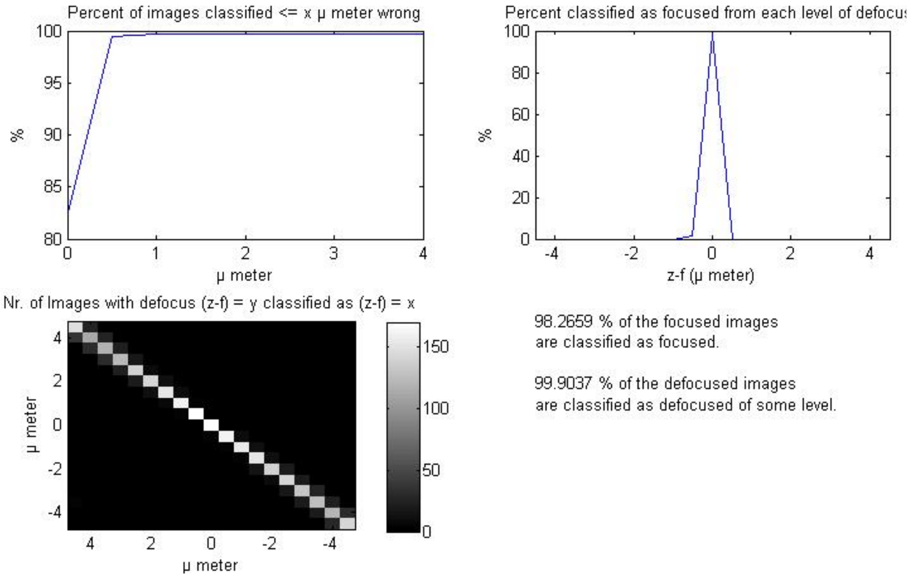

Figure 4.8: Results from Training/Testing on Database3 using the all features.

the same method worked well on both databases. There was a difference, though, when using the reduced feature set. The features in this set were selected when testing and training on Database1. The ambition was that this selection would improve the SVM performance on any data but it is easy to conclude from Fig. 4.1 that this is not the case. The results from the features selection section (Section 3.4) are hence not good to use on an arbitrary database. Also, when using the segmentation independent features, there is a difference from when Database1 was used. This time the performance only decreased with 17%, and these results border on being good enough to use. Detailed results when using all features are in Fig. 4.8, and when using segmentation independent features in Fig. 4.9

If the gain in computational time is big enough when using only segmentation independent features, one could adopt a two step approach, where, if the first step was not quite right, another one is taken. Looking at Fig. 4.9 there is over 99% chance of being within $1.5\mu m$ from focus. From there the accuracy is very good, even if only the segmentation independent features were used. This is illustrated in Fig. 4.10.

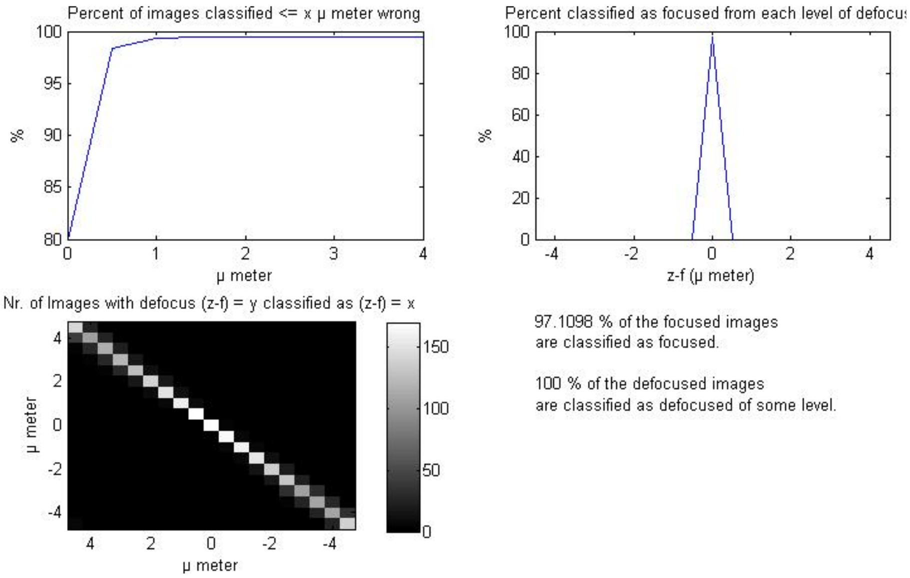

Figure 4.9: Results from Training/Testing on Database3 using segmentation independent features. The performance is not as good as when using all features but the difference is small. Note that no images are falsely classified as focused.

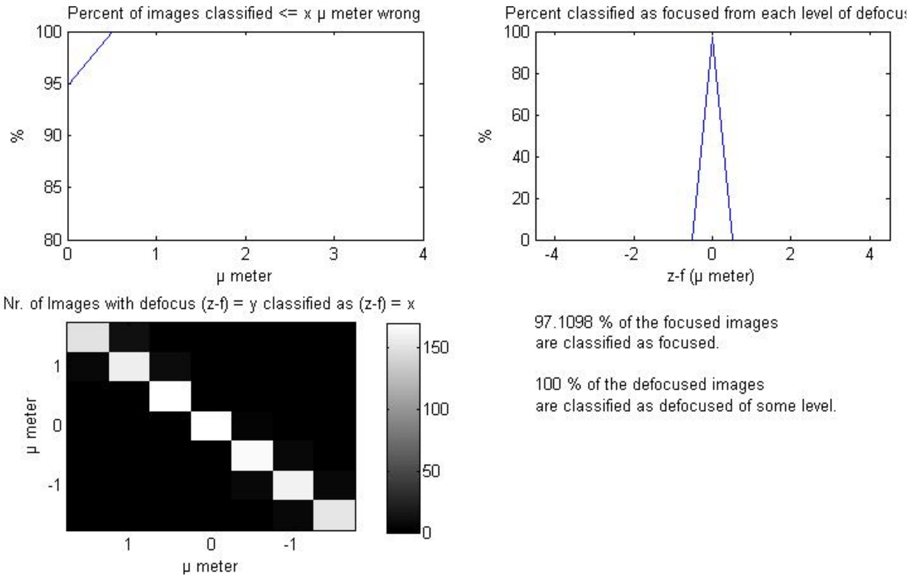

Figure 4.10: Results from Training/Testing on images with $|z - f| \leq 1.5\mu m$ from Database3 using segmentation independent features. Note how 94% of the images are classified correctly and how virtually no images are misclassified more than $0.5\mu m$.

Chapter 5

Conclusions and future work

This thesis has showed that it is possible to create an absolute focus metric in blood smear images. A defocused image's distance to focus, as well as direction of defocus, can be calculated with good accuracy. Using test data with distances to focus between $-4.5\mu m$ and $4.5\mu m$, 99.5% of the images's distances to focus were estimated less or equal to $0.5\mu m$ wrong while over 85% were estimated completely correctly. A $0.5\mu m$ defocus is borderline to what the human eye perceives as defocused. Also, vibrations and plays in the machine cause noise of almost this level. A $0.5\mu m$ misclassification does therefore not necessarily result in that the next image is defocused. Using test data with distances to focus between $-2.5\mu m$ and $2.5\mu m$, 99.9% of the images's distances to focus were estimated less or equal to $0.5\mu m$ wrong and 94% were estimated completely correctly. The accuracy of the method thus increases when the images are only slightly defocused.

This method opens up for new ways of capturing a focus image in an auto focus system. After capturing a test image, the computer can tell the microscope how far to move in order to make sure that the next image is well focused. If the distance is slightly misjudged an additional step with better precision can be taken.

In order to get the best accuracy from this method the images have to be segmented and features calculated for the different parts of the image. However, a good accuracy can be achieved without using these segmentation depending features. Doing this would speed up the image evaluation process significantly. The exact time gain remains however to be investigated.

Concerning the data used for training, the larger Database3 worked better than the smaller Database1 when classifying images from Database2. This can be explained by the fact that it is larger, but also because Database1 was collected with another machine that the two others. Therefore to train the SVM with data from the DM8 and then classify images from the DM96 does not seem to work, but the opposite remains to be tested.

When it comes to future work, there is room for improvements. The following are suggestions on further work.

Investigate method computational time for different selections of features, and decide if it is worth the extra time is takes to make the segmentation and extract the segmentation depending features. If the segmentation depending features are to be used a robust segmentation that deals with images without a WBC needs to be made. Then continue the feature reduction analysis and conclude which groups of features are necessary and vital for good performance. These features should give good results on different databases, not only on the one used when selecting them. Also, make sure that a SVM trained on data from one machine can classify data from another machine of the same kind successfully. Further, fine tune features, in particular the Wavelet based contrast feature, calibrate C and γ more carefully and investigate how many level of defocus should be included in the data used for training.

Bibliography

[1] L. Jinlong, Z. Chau and S. Qingyun. *Estimating the amount of defocus through a wavelet transform approach* Elsevier 2003

[2] J. Kautsky, J. Flusser, B. Zitová and S. Šimberová *A new wavelet-based measure of image focus* Elsevier 2003

[3] L-C. Böiers *Lectures on Optimization* KFS 2004

[4] N. Ross *In vitro characterisation of the relationship between plasma glucose level and the refractive index of the lens of the eye: Chapter 3, Image processing* University of the Witwatersrand, Johannesburg

[5] N. Cristianini and J. Shawe-Taylor *An Introduction to Support Vector Machines* Cambridge University Press 2000

[6] F. Groen, I. Young, and G. Ligthart *A Comparison of Different Focus Functions for Use in Autofocus Algorithms* Department of Applied Physics, Delft University of Technology, 2600 GA Delft, The Netherlands 1984

[7] N. Otsu *A threshold selection method from gray-level histograms* IEEE Transactions on Systems, Man, and Cybernetics, 9(1):62-66, 1979.

[8] A. Karlsson *Area-Based Active Contours with Applications in Medical Microscopy: Paper 1, chapter 3* Licentiate Thesis in Mathematical Sciences 2005:4 ISSN 1404-028x

[9] S. S. Reddi, S.F. Rudin, and H.R. Keshavan *An optimal multiple threshold scheme for image segmentation.* IEEE Transactions on Systems, Man, and Cybernetics, 14(4):661-666, 1984.

[10] T. Kurita, N. Otsu, and N. Abdelmalek *Maximum likelihood thresholding based on population mixture models* Pattern Recognition, 25(10): 1231-1240, 1992

[11] C. Gonzales and E. Woods *Computer vision - a modern approach* Pearson Education Inc. 2003, ISBN 0-13-085198-1

[12] H. Chih-Wei, C. Chih-Chung, and L. Chih-Jen *A Practical Guide to Support Vector Classification* Department of Computer Science and Information Engineering National Taiwan University

[13] R. K. Bryll *System and method for single image focus assessment* United States Patent Application Publication, Pub. No.: US 2006/0204121 A1. Pub. date: Sep 14 2006

Appendix A

List of features

Content features

Nr:	Feature	Layer	Impact
1	Mean intensity, whole image	r	
2	Mean intensity, whole image	g	
3	Mean intensity, whole image	b	
4	Mean intensity, whole image	mean	
5	Mean intensity, background	r	
6	Mean intensity, background	g	
7	Mean intensity, background	b	
8	Mean intensity, background	mean	
9	Mean intensity, RBCs	r	
10	Mean intensity, RBCs	g	
11	Mean intensity, RBCs	b	
12	Mean intensity, RBCs	mean	
13	Mean intensity, WBCs	r	
14	Mean intensity, WBCs	g	
15	Mean intensity, WBCs	b	
16	Mean intensity, WBCs	mean	
17	Standard deviation, whole image	r	
18	Standard deviation, whole image	g	
19	Standard deviation, whole image	b	
20	Standard deviation, background	r	
21	Standard deviation, background	g	
22	Standard deviation, background	b	
23	Standard deviation, RBCs	r	
24	Standard deviation, RBCs	g	
25	Standard deviation, RBCs	b	

26	Standard deviation, WBCs	r
27	Standard deviation, WBCs	g
28	Standard deviation, WBCs	b
29	Number of pixels occupied by RBCs	binary
30	Number of pixels occupied by WBCs	binary

Global contrast features

Nr:	Feature	Layer	Impact
31	Waveletbased measure	mean	
32	Blurvalue	r	
33	Blurvalue	g	
34	Focusvalue	r	
35	Focusvalue	g	
36	Focusvalue	b	
37	Derivation based measure	r	
38	Derivation based measure	g	
39	Derivation based measure	b	
40	Derivation based measure	mean	
41	Vollath's F4	r	
42	Mix of Vollath's F4 and Vollath's F5	r	
43	Vollath's F4	g	
44	Mix of Vollath's F4 and Vollath's F5	g	
45	Vollath's F4	b	
46	Mix of Vollath's F4 and Vollath's F5	b	
47	Vollath's F4	mean	
48	Mix of Vollath's F4 and Vollath's F5	mean	

WBC contrast features

Nr:	Feature	Layer	Impact
49	Focusvalue	r	
50	Focusvalue	g	
51	Focusvalue	b	
52	Derivation based measure	r	
53	Derivation based measure	g	
54	Derivation based measure	b	
55	Derivation based measure	mean	
56	Vollath's F4	r	
57	Mix of Vollath's F4 and Vollath's F5	r	
58	Vollath's F4	g	

59	Mix of Vollath's F4 and Vollath's F5	g
60	Vollath's F4	b
61	Mix of Vollath's F4 and Vollath's F5	b
62	Vollath's F4	mean
63	Mix of Vollath's F4 and Vollath's F5	mean
64	Size of the box around the WCB in which the WBC contrast features are calculated	binary

Directional feature

Nr:	**Feature**	**Layer**	**Impact**
65	Dirvalue	b and g	

Appendix B

Databases

Database1

4902 images in 248 series collected with the DM8. Each series consist of 19 images of the same object, with distances to focus

$$(z - f) = [4.5, 4, \dots, 0.5, 0, -0.5, \dots, -4, -4.5]\mu m.$$

Series selected from different laboratories.

Database2

12825 images in 513 series collected with the DM96. Each series consist of 25 images, with distances to focus

$$(z - f) = [6, 5.5, \dots, 0.5, 0, -0.5, \dots, -5.5, -6]\mu m.$$

Series selected from the same laboratory.

Database3

13129 images in 619 series collected with the DM96. Each series consist of 19 images, with distances to focus

$$(z - f) = [4.5, 4, \dots, 0.5, 0, -0.5, \dots, -4, -4.5]\mu m.$$

Series selected from different laboratories.

Appendix C

Feature tables

Reduce (error v. 0.170)

	r	g	b	o
brightness tot			1	
brightness bk		1		
brightness rbc	1	1		1
brightness wbc		1	1	
variance total	1	1	1	■
variance bk	1		1	■
variance rbc		1		■
variance wbc	1			■
area rbc	■	■	■	1
area wbc	■	■	■	
wavelet	■	■	■	1
blurvalue		1	■	■
focusvalue			1	■
derivation	1	1	1	1
vollath F4		1	1	
mix F4 and F5			1	
focusvalue wbc	1	1	1	■
derivation wbc	1	1	1	1
vollath4 wbc	1		1	1
oscar wbc	1			
size of wbc	■	■	■	1
dirvalue	■	■	■	1
Sum:	9	10	11	8
			=	38

Add (error v. 0.178)

	r	g	b	o
brightness tot	1	1	1	1
brightness bk	1	1	1	1
brightness rbc	1	1	1	1
brightness wbc	1	1	1	1
variance total		1	1	■
variance bk	1	1	1	■
variance rbc		1		■
variance wbc		1		■
area rbc	■	■	■	1
area wbc	■	■	■	
wavelet	■	■	■	1
blurvalue	1	1	■	■
focusvalue	1	1	1	■
derivation	1	1	1	1
vollath F4		1	1	1
mix F4 and F5	1	1	1	1
focusvalue wbc	1		1	■
derivation wbc			1	
vollath4 wbc				
oscar wbc				
size of wbc	■	■	■	1
dirvalue	■	■	■	1
Sum:	10	13	12	11
			=	46

Random (error v. 0.183)

	r	g	b	o
brightness tot	1		1	
brightness bk	1	1		
brightness rbc	1	1		1
brightness wbc		1	1	1
variance total		1	1	■
variance bk	1	1	1	■
variance rbc		1		■
variance wbc			1	■
area rbc	■	■	■	1
area wbc	■	■	■	1
wavelet	■	■	■	1
blurvalue	1	1	■	■
focusvalue	1	1	1	■
derivation	1	1	1	1
vollath F4				
mix F4 and F5		1		
focusvalue wbc		1	1	■
derivation wbc			1	
vollath4 wbc			1	
oscar wbc	1			1
size of wbc	■	■	■	1
dirvalue	■	■	■	1
Sum:	8	11	10	9
			=	38

🟨 "=segmentation independent" r=red, g=green, b=blue, o=other

■ "=n/a"

Figure C.1: The best features from the reduction and addition analysis as well as the best 38 from the random analysis approach. Note that the features in all tables are divided evenly across the three color layers.

Appendix D

Feature sets

Set A = $[2, 5, 10, 14, 19, 20, 24, 27, 29, 31, 32, 36, 40, 46, 47, 51, 54, 61, 62, 65]$

Set B = $[14, 15, 18, 19, 20, 22, 24, 29, 31, 32, 33, 34, 35, 36, 38, 40, 50, 51, 54, 65]$

Set C = $[10, 14, 15, 17, 18, 19, 20, 22, 29, 32, 36, 37, 38, 49, 51, 55, 59, 62, 64, 65]$

Set D = $[5, 13, 14, 16, 20, 21, 22, 29, 31, 33, 34, 35, 36, 37, 38, 39, 40, 41, 45, 65]$

Set E = $[14, 20, 29, 36, 65]$

Set F = $[14, 20, 22, 29, 36, 38, 65]$

Set G = $[5, 10, 14, 15, 18, 19, 20, 22, 29, 31, 32, 33, 34, 35, 36, 38, 40, 51, 62, 65]$