From the Institute for Neuro- and Bioinformatics
of the University of Lübeck
Director: Prof. Dr. rer. nat. Thomas Martinetz

Machine Vision for Inspection and Novelty Detection

Dissertation
for Fulfillment of Requirements for the Doctoral Degree
of the University of Lübeck
from the Department of Computer Sciences/Engineering

Submitted by
Fabian Timm from Pinneberg

Lübeck 2011

First referee:	Prof. Dr.-Ing. Erhardt Barth
Second referee:	Prof. Dr. rer. nat. Bernd Fischer
Date of oral examination:	2011/11/09

Approved for printing. Lübeck, 2011/11/14

Machine Vision for Inspection and Novelty Detection

Fabian Timm

Wo nur habe ich meinen Schlüssel hingelegt? Nach langem Grübeln können wir uns dennoch nicht an den Ort des Schlüssels erinnern und beginnen mit der Suche. Häufig endet diese Suche nach kurzer Zeit erfolgreich, manchmal hat man jedoch das Gefühl, die Nadel im Heuhaufen zu suchen.

Auch wenn diese Suchaufgabe bereits schwierig erscheint, gibt es noch weit komplizierere Suchaufgaben – wenn beispielsweise das Suchobjekt teilweise oder gänzlich unbekannt ist. Solche Suchaufgaben finden wir im alltäglichen Leben, besonders aber in der industriellen Produktion, wo lange und komplexe Produktionsanlagen verwendet werden. Zur Qualitätssicherung werden dort verschiedene Inspektionssysteme eingesetzt, die beispielsweise Druck, Temperatur oder Feuchtigkeit messen und einen Alarm auslösen, sobald diese Messdaten nicht den Anforderungen genügen. Treten allerdings sehr subtile, optische Defekte auf, wird eine manuelle Kontrolle jedes einzelnes Produktes vorgenommen. Eine manuelle Kontrolle ist besonders dann unumgänglich, wenn unbekannt ist, welche Defekte überhaupt auftreten können; in diesen Fällen werden Experten an zahlreichen fehlerfreien Beispielen und einigen wenigen fehlerhaften trainiert. In dieser Arbeit beschreiben wir optische Mustererkennungsysteme die genau diese Suchaufgabe – die Detektion von beliebigen und unspezifizierten Abweichungen von fehlerfreien Produkten – automatisch lösen können.

Ein Mustererkennungsystem besteht in der Regel aus drei Ebenen: Bildvorverarbeitung, Extraktion von charakteristischen Bildeigenschaften und Klassifikation dieser Eigenschaften. Im theoretischen Teil dieser Arbeit stellen wir für jeden der drei Bereiche neue Methoden vor, im praktischen Teil zeigen wir dann deren Anwendung in unterschiedlichen Mustererkennungsystemen mit besonderem Schwerpunkt auf Inspektionssysteme zur optischen Qualitätsprüfung.

Die Bildvorverarbeitung ist für alle nachfolgenden Schritte innerhalb eines Mustererkennungssystems von großer Bedeutung, denn Ungenauigkeiten, beispielsweise bei der Bildverbesserung oder der Extraktion von relevanten Bildbereichen, können anschließend nur schwer kompensiert werden. Wir präsentieren zwei neue Verfahren zur Bildvorverarbeitung. Das eine Verfahren kompensiert inhomogene Ausleuchtungen und verbessert das

Bild derart, dass kein Beleuchtungsgradient mehr erkennbar ist und die Bildränder nahezu optimal extrapoliert werden. Das andere Verfahren dient zur Extraktion von runden Objekten im Bild. Wir zeigen, dass auch bei extremen Bildrauschen, Verdeckung und Reflektionen die runden Objekte mit höher Präzision detektiert werden können. Diese Methode kann als vorverarbeitender Schritt bei der Inspektion von Schweißpunkten, p-Elektroden von LEDs oder auch zum Eye-Tracking eingesetzt werden.

Zur Extraktion von charakteristischen Bildeigenschaften gibt es, prinzipiell, zwei unterschiedliche Ansätze – entweder werden die unverarbeiteten Intensitäten der relevanten Bildregion zeilen- oder spaltenweise in einen Vektor geschrieben oder problemspezifische Bildmerkmale berechnet, wie beispielsweise Form oder Textur. Wir stellen vier neue Methoden zur Berechnung von charakteristischen Bildeigenschaften vor. Für die Inspektion von Schweißpunkten berechnen wir Statistiken über die Form und Gestalt von Zusammenhangskomponenten in einer Reihe von Binärbildern; dabei verwenden wir einerseits Eigenschaften wie Kompaktheit, Ausdehnung oder Irregularität und anderseits Fourier Deskriptoren. Für die Inspektion von p-Elektroden in LEDs ermitteln wir Statistiken über die Grauwerte innerhalb eines radialen Gitters und über die Grauwerte auf radialen Projektionen. Bei der Detektion von subtilen Fehlern in Texturen verwenden wir die Verteilung von lokalen Gradienten; dazu modellieren wir die Verteilung der Beträge der Gradienten durch eine Weibull-Verteilung und schätzen ihre Parameter. Wir konnten feststellen, dass sich defekt-freie Bildregionen im Raum der Weibull-Parameter in einer bestimmten Region befinden, wohingegen defekte Bildregionen signifikant davon abweichen.

Im letzten Schritt eines Mustererkennungssystems werden die extrahierten Bildeigenschaften klassifiziert und, im Falle einer industriellen Inspektion, entschieden, ob das Produkt fehlerhaft oder fehlerfrei ist. Gewöhnlich werden dazu Methoden eingesetzt, die den Merkmalsraum so in zwei Halbräume (fehlerhaft vs. fehlerfrei) einteilt, dass der Klassifikationsfehler auf Trainingsdaten und Testdaten minimiert wird. Hierfür müssen allerdings ausreichend Daten von beiden Klassen vorliegen – besonders in industriellen, aber auch in medizinischen Anwendungen können selten ausreichend negative Daten akquiriert werden. In dieser Arbeit verwenden wir deshalb Methoden zum Lernen der positiven Beispiele und zur anschließenden Detektion von Ausreißern. Somit kommen wir nahezu ohne negative Beispiele aus, können aber gleichzeitig beliebige und auch unbekannte negative Beispiele erkennen. Wir haben einen neuen, simplen Algorithmus zur Lösung dieses Lernproblems entwickelt, der vergleichbare Ergebnisse hinsichtlich Performanz und Geschwindigkeit zu aktuellen Lösungsalgorithmen erzielt – unser Algorithmus kann allerdings ohne besondere Bibliotheken, ausschließlich mit Standard-Operatoren und in wenigen Zeilen implementiert werden.

Im zweiten Teil dieser Arbeit beschreiben wir Musterkennungssysteme für verschiedene, industrielle Anwendungen und verwenden die im ersten Teil beschriebenen Methoden zur Vorverarbeitung, Merkmalsextraktion und Klassifikation. Außerdem zeigen wir, dass unsere Methoden auch in nicht-industriellen Anwendungen verwendet werden können.

Wenn ich mein Leben noch einmal leben könnte,
würde ich mehr Fehler machen.
Ich würde bis zum Äußersten gehen.
Ich würde alberner und verrückter sein
und würde mehr Chancen wahrnehmen.
Ich würde mehr unternehmen,
würde mehr Berge besteigen,
in mehr Flüssen schwimmen
und mehr Sonnenuntergänge beobachten.

Nadine Stair (im Alter von 85 Jahren)

Acknowledgements

"*Pursuing a PhD is like running a marathon that seems to never end.*“, a friend of mine once said. I think it's even more than that; pursuing a PhD is like doing a triathlon—you must prove to be good in different disciplines, you must have sufficient stamina, and you must train a lot to be well prepared for the race. Obviously, you cannot do this completely on your own—you rather need to have a team that supports you. There are many people that supported me while I was working on this thesis and that I would like to thank for.

I thank my coach and supervisor, Erhardt Barth, for his technical and moral support at all times. I thank Thomas Martinetz for giving me the opportunity to pursue my PhD, for his fruitful discussions on machine learning, and for the great research environment at the Institute for Neuro- and Bioinformatics.

Commonly, a PhD thesis can only cover a fraction of things you have done during your PhD and only few people know about everything else. I thank my teammate, Sascha Klement, for many intensive discussions on machine vision and machine learning, but also for all the other things we have gone through during my PhD as well as during undergraduate study.

I thank Kai Labusch, with whom I worked together on novelty detection methods that resulted in the work described here.

Michael Dorr and Martin Böhme helped me to improve my programming skills as well as my writing and design skills, and they were the right persons to go to when general advice was required or I felt like talking to people that have already gone through the same ups and downs.

Many students have worked with me during their undergraduate theses—I thank Mike Wellner, Martin Lechner, Kristian Ehlers, and Nadine Traulsen for their great work.

This thesis would not have turned out the same without several real-world applications and I thank Chrisitian Arlt and Jürgen Klein from Philips GmbH, GTD Mechanisation, Aachen, for great collaboration and financial support over the last few years.

My deepest thanks go to my family for their personal support and their patience over the last decade. Finally, I thank you, Anke!

Contents

Introduction

Now where did I put my watch? Everybody knows the feeling of searching desperately for something particular—a key, remote control, or some other small object. Since we have forgotten where we put that object, we start searching everywhere—in the house, in the jacket, or in the pocket. Sometimes this search seems like looking for a needle in a haystack that contains several objects similar to the one we are searching for. Fortunately, we usually find the desired object sooner or later.

Searching for a particular object is already difficult, but it can even be more difficult if we start searching for an unknown object; for example, finding a software bug can sometimes take a couple of hours or even days, since we do not know what exactly may cause the problem; in medicine, the pathogen can often be identified within few days, but searching for the source of the pathogene sometimes takes weeks or months, as it is currently the case for the mutated *E. coli*; in industrial manufacturing a large production line often contains various machines and a defective product may be caused by a single machine or by subtle changes at several machines. Therefore, a huge effort is being made on monitoring and controlling production lines to provide the desired quality, especially for manufacturing processes that require very high quality, such as in the automotive industry. Even though we can monitor manufacturing by analysing data from various sensors, such as pressure, temperature, or humidity, defects that affect the appearance can only be measured by optical inspection, i.e. machine vision systems.

Various machine vision systems for optical inspection during manufacturing have been developed; for example, we can track products through manufacturing by high-performance digit recognition or data matrix code recognition; we can automatically analyse geometry and shape to ensure that parts can be joined properly; we can verify colour to guarantee consistent appearance over time, which can be very difficult, for instance, in the case of metallic paint.

However, sophisticated machine vision problems such as the inspection of highly structured surfaces, where defects are specified weakly and the surface shows diverse structures, still require a manual inspection. Humans can even detect defects if they have only seen a single example of a defective sample before, and they are able to identify defects that differ from any defect example they have seen before. Even though manual inspection yields acceptable performance and robustness, it is not only costly and time-consuming, but the production is no longer fully reproducible due to variations in human performance.

Four major issues have to be solved for the development of a machine vision system for surface inspection. First, we must develop methods for detecting surface defects that are only roughly specified, but at the same time we must ensure that unknown types of defects are detected as well. Therefore, we have to compute characteristic features that can capture small deviations from the original surface, and we then have to classify these features to identify known and unknown defects. Second, we have to cope with highly structured images such as metallic or highly reflective surfaces and, third, image noise makes an automated inspection of such highly structured images even more difficult. Fourth, defects are rare and only weakly labelled; to detect every possible defect we can therefore only perform novelty-detection based on characteristic image properties of defect-free samples.

For most applications, a machine vision system consists of three stages after the image has been acquired: First, the image is preprocessed to reduce noise, improve image quality, or extract regions of interest, for example; second, characteristic image features, often specifically designed for the application, are extracted; finally, the objects are classified to distinguish defective image regions from defect-free regions.

In this thesis, I will present novel methods for each of these three stages, and I will demonstrate their performance on benchmark datasets as well as real-world applications. The methods presented in this thesis were mainly developed for industrial applications, such as the detection of defects in texture images or the inspection of welding seams, but I will show that some of these methods can successfully be applied to other areas as well. This thesis is organised into two parts—theory and application. After a brief introduction of machine vision systems for inspection and novelty detection and a brief specification of the challenges faced in developing a machine vision system, especially in the case of images with varying texture, the theory part describes methods for image processing, feature extraction, and novelty detection. At the beginning of each chapter, I will present a map that shows the methods I have developed and that indicates at which stage of a machine vision system the methods of the current chapter are used.

I will introduce three novel methods for image preprocessing; two of these deal with accurate, robust, and efficient centre detection of (semi-) circular objects and can be used in a wide range of applications such as inspection of LEDs or eye tracking. The third method is for removing inhomogeneities in texture images; inhomogeneous illumination is a major problem for automatic surface inspection, and it may cause many false negatives, especially in the case of highly structured or textured images.

I have developed four novel methods for computing characteristic image features. The first method computes so-called specularity features, which capture shape statistics of specular reflections and can, for example, be used for surface inspection. The second method is based on Fourier descriptors and is closely related to the specularity features; instead of computing specifically designed shape properties such as compactness,

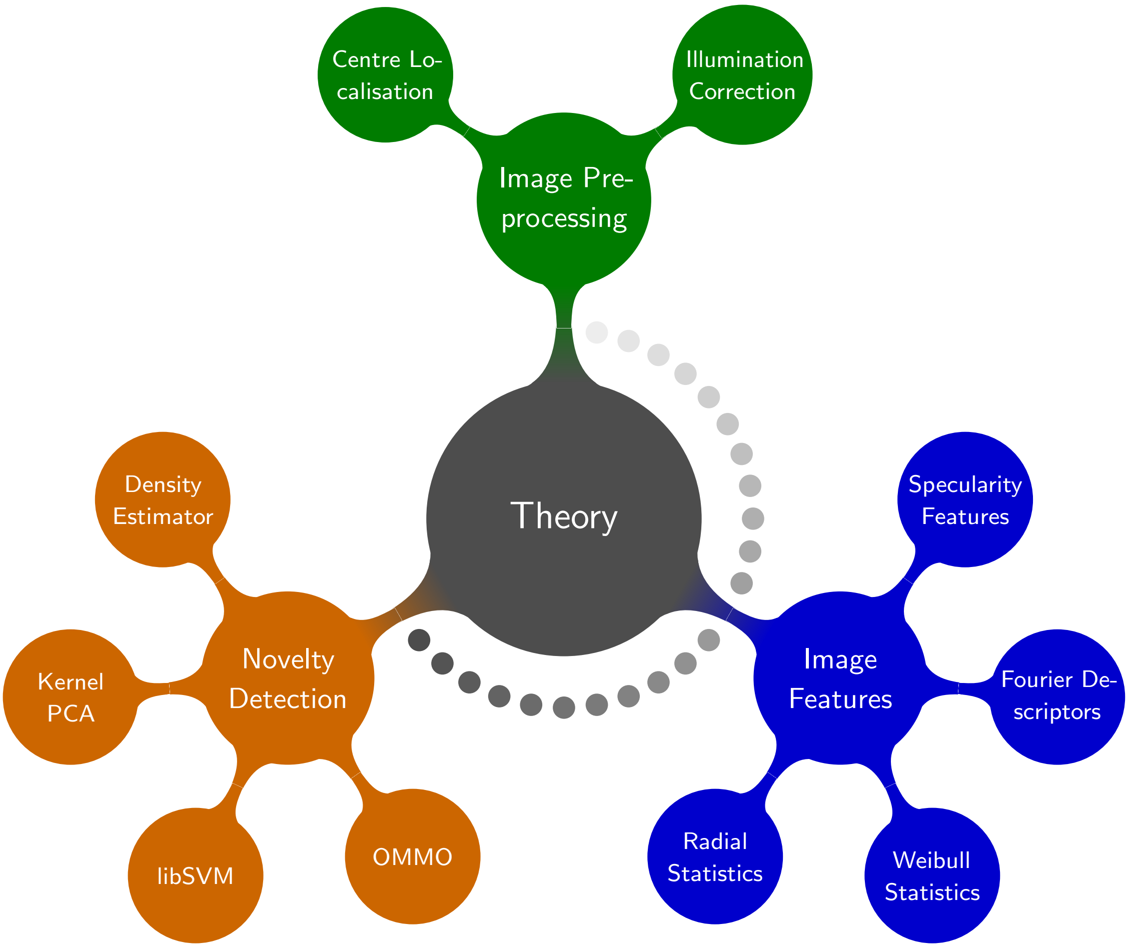
irregularity, or extent, I use the Fourier descriptors of the object's boundary and combine them by first order statistics. The third method captures statistics of raw intensities by placing a radial grid over the image; it can, for example, be used for detecting subtle defects in images of light-emitting diodes. The fourth method employs the local distribution of image gradient orientations and computes the parameters of a Weibull fit of this distribution. These Weibull parameters have recently been used to classify image content; in contrast, I demonstrate that Weibull features can also be applied to describe local image characteristics and detect arbitrary deviations in texture images.

Since it is often impossible to acquire sufficiently many negative samples, I focus, for classifying image features, on novelty-detection approaches rather than standard classification approaches. Moreover, studies have proven that standard classification methods that try to separate two classes with minimum error yield poor performance if the classes are highly unbalanced. Especially in medical and industrial applications, only few or even no negative samples are available; in these cases, novelty detection approaches significantly outperform standard classification approaches. Here, I introduce a novelty-detection approach that is based on the maximum-margin principle and that can be implemented within only a few lines of code. The novel approach performs comparable to sophisticated state-of-the-art toolboxes and can also be applied to large-scale applications with high dimensions. Because of its simplicity even practitioners or beginners in the field of machine learning, can implement the novel algorithm—without any specific knowledge in optimisation theory.

In the applications part, I demonstrate that the novel methods, presented in the theory part, can successfully be applied to real-world applications. First, I show that the specularity features as well as the statistical Fourier descriptors yield high performance for the problem of welding seam inspection. Second, I use the centre localisation approach to detect regions of interest in images of light-emitting diodes, and I use the radial-feature statistics to detect small, subtle defects in p-electrodes of LEDs. Third, I show that local Weibull features can be applied to the problem of defect detection in texture images; I demonstrate that this novel approach is neither limited to a particular type of texture nor to a particular defect type but yields high accuracy for a diverse spectrum of textures and defects. Finally, I show that some of our methods can successfully be applied to non-industrial applications such as the problem of accurate eye centre localisation, which can be used as part of an eye tracker.

Some of the results presented in this thesis were developed as part of a group effort and many of the proposed methods have already been integrated into industrial systems. Therefore, I will, at the beginning of the corresponding section, identify which contributions are my own. For the sake of consistency, I will use the pronoun "we" throughout, even when describing results that are solely my own.

An overview of the methods used in this thesis is given below; all methods have newly been developed except for the kernel density estimator, kernel PCA, and libSVM:



Part I.

Machine Vision Theory

1. A Brief Overview

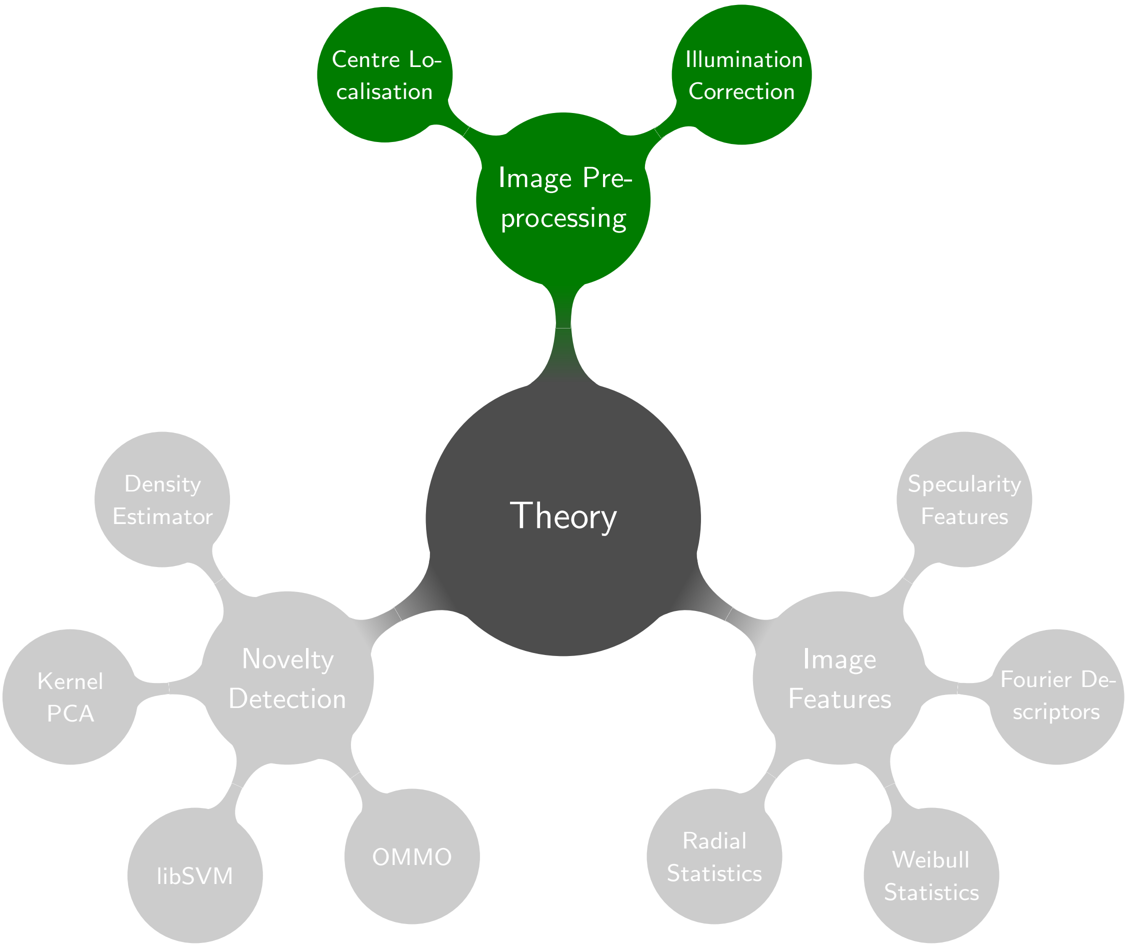
Over the past 30 years, texture analysis has become an important task in machine vision and can roughly be divided into three groups according to the application: texture classification, texture segmentation, and texture synthesis. We can recognise different types of texture, but we can hardly describe the specific properties of texture; hence, it is very difficult to define the term "texture" at all. Since we do not want to elaborate on methods for texture analysis here, we refer to [106], for instance.

In this thesis, we focus on machine vision systems for the detection of small and subtle defects in texture images. This can be interpreted as both texture classification and texture segmentation. Since we measure local deviations from the original texture, we can not only detect a defective image, but we can also localise the defect, which can then be used for further analysis and cause studies, e.g. statistical maps of defective regions. However, we have to make a compromise between accuracy of defect localisation and computation time. Moreover, the machine vision system must yield a good balance between high detection rate for defects on the one hand and at the same time it must not yield many false negatives, which is very challenging, especially when the defect-free texture varies strongly.

Even though many methods for the detection of defects in texture images have been proposed—surveys can be found in [115, 62, 61, 83], for example—two major issues remain. First, these methods have often been developed for a particular application such as the inspection of TFT panels or the inspection of semi-conductor components and are therefore highly adapted to the particular dataset, which is, unfortunately, not made publicly available; hence, a fair comparison of recent approaches to the problem of texture defect detection is missing. Second, the definition of the problem varies significantly; in some cases, texture defect detection is treated as unsupervised learning of arbitrary defective regions in texture images in the absence of negative samples; in other cases, it is treated as a supervised learning task with sufficiently many negative samples.

In this thesis, we define the problem of defect detection in texture images with the following challenges: (i) defect-free regions show a certain type of texture that can significantly vary, (ii) defective regions show a texture that can be arbitrary, but that is different compared to defect-free regions, (iii) only few negative samples are available, (iv) negative samples contain small, subtle, and weakly labelled defects. Since we only have sufficiently many defect-free samples, we apply learning schemes that try to describe the class of all positive samples and that detect every negative sample as an outlier. Moreover, we will apply our methods not only to real-world applications, but also to a benchmark dataset that has recently been published by the company Robert Bosch GmbH.

2. Image Preprocessing



Parts of this chapter are joint work with others. Sascha Klement and I came up with idea of removing inhomogeneous illumination in texture images; we contributed approximately equally to refining that idea and Sascha Klement did most of the implementation. The idea for accurate centre localisation, the derivation of the methods, and their implementation were completely made by myself. Some of the work described in this chapter has been previously published in [57, 99, 100]. Parts of the proposed methods have been integrated into industrial applications (see Part II) and commercial software products [3].

2.1. Introduction

Image preprocessing has become a large field of research with a wide range of methods for different applications. In general, image preprocessing is performed after retrieving image data, and it strongly influences further feature extraction and classification; hence, techniques for image preprocessing must be efficient, but accurate. Preprocessing methods can be separated into several groups according to their semantic level, e.g. pixel-based or content-based.

Modifications of brightness, contrast, or dynamic range are most frequently used to enhance a single image; sophisticated techniques such as image registration, image warping, or image stitching must be used to process a sequence of images, for example when analysing satellite image data or when using stereo vision systems. When the image is composed of various parts, the region of interest (ROI) must be detected before characteristic image features can be computed; in some cases an accurate detection of the ROI can even be more complicated than feature extraction and classification.

In this chapter, we will introduce approaches for two different areas—accurate ROI detection and image enhancement. First, we propose two novel methods for the accurate localisation of (semi-) circular objects; the first method is based on the distribution of image gradients and can be computed very efficiently; the second method determines the optimal centre by analysing the intensity variations radially. For both we mathematically formulate a cost-function; for the gradient-based approach we also derive an incremental algorithm. A comprehensive comparison with state-of-the-art methods is presented in Part II, where we demonstrate that our methods can successfully be applied to real-world applications.

Second, we propose a novel approach for the problem of illumination correction and optimised image stitching, which can be used for virtual material design or as preprocessing for surface inspection. With standard filtering images will show artefacts at the image borders—mostly visible in the case of image stitching. Our novel method not only removes inhomogeneities accurately without boundary artefacts, but it also qualifies for realtime applications. The novel approach is presented within a framework that theoretically allows any type of boundary conditions, however, we will demonstrate that in most cases linear or polynomial extrapolation already yields accurate results.

2.2. Accurate Centre Localisation

The problem of object detection and in particular the problem of detecting the centre of a (semi-) circular object has been widely addressed in various research fields such as industrial imaging (surface inspection), medical imaging (cell tracking), or vision science (eye tracking), for example. We want to address the problem of centre localisation with a special focus on robustness. Strong distortions such as noise, occlusions, reflections, or blur complicate an accurate detection of the object's centre—sometimes the centre is hard to detect even manually (see Figure 2.1).

One of the well-known approaches for the detection of circles is the Hough transform [47, 29, 56, 75]. The Hough transform approach is a voting-based technique and motivated by the idea that each sample, e.g. a contour point obtained by edge detection or a point in a binary image, contributes to a globally consistent solution. Over the past 35 years, several modifications have been proposed to improve the Hough transform in terms of accuracy and efficiency [49, 56, 53]; a good survey on Hough transforms was presented by Illingworth and Kittler [48]. However, several comparative studies show that the information about edges in an image is insufficient to detect circles accurately, especially in the presence of noise [8, 117]. To overcome this problem, Ceccaralli et al. proposed a method that avoids the detection of edges, but tries to detect a circular object by using a template matching mechanism that retains the information about pixel intensity [14]. In particular, template matching is performed between the direction of the gradient at each image position and the gradient direction of an ideal circle whose radius varies in a given interval. However, template matching is often computationally inefficient and the performance depends on the quality of the template and, especially, the noise level.

Furthermore, isophote properties have been used for object detection [67] and eye centre localisation [108]. The latter employs a voting scheme like the Hough transform, where, for every pixel, the centre of the osculating circle of the isophote is computed from smoothed derivatives of the image brightness. However, these voting-based approaches are also inefficient and lead to the problem of analysing the voting-space, which may contain many local maxima; post-processing techniques must then be applied to identify a single maximum.

We propose two novel approaches that accurately localise the centre of a (semi-) circular object, even in case of strong noise and low contrast images; the first approach is based on image gradients with a novel objective function, which has to be maximised and for which we develop a simple and fast iterative algorithm; the second approach

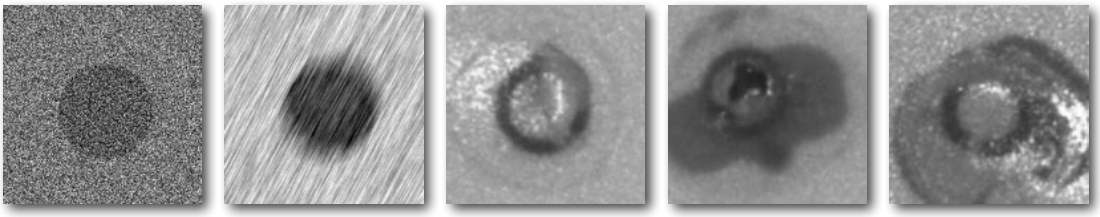


Figure 2.1.: Artificial and real-world images of circular objects with various image distortions such as noise, motion blur, occlusions, or reflections. In these scenarios, a method for accurate centre localisation must be extremely robust.

is inspired by the concept of isophotes, for which we derive an objective function based on the definition of radial symmetry. Geometrically, given the optimal centre, the intensities, considered as radially, of a circle vary only slightly; whereas for incorrect centres, the radial intensities vary significantly. Therefore, the centre is the location where the mean variation of intensities over several radii reaches its minimum. In this section, we apply the two novel approaches to synthetic images and we compare to an approach based on the Hough transform; in Part II, we apply the approaches to the problem of LED inspection and eye centre detection.

2.2.1. Edge-based approach

As a reference method, we use the Hough transform [47, 29] for the detection of a circular object and apply the following steps to obtain the object's centre (see Figure 2.2): (i) edge detection to obtain the object's contour, (ii) computing the accumulator array in a predefined space, and (iii) evaluating the maximum of the accumulator array. Since we assume that the circular object is completely located inside the image, the accumulator array is evaluated for $c_x \in [1, W]$, $c_y \in [1, H]$, and $r \in [1, 0.5 \times \min(W, H)]$. The only remaining parameter is the accuracy in each dimension of the accumulator, which we set to 0.5 to achieve sub-pixel accuracy. We apply the Canny algorithm with standard parameters to obtain contour points (Gaussian smoothing with $\sigma = 1$ and size 5×5, the high threshold t is automatically computed from the cumulated sum of the distribution of gradient magnitudes, and the low threshold is computed by $0.4 \times t$). We search for the total maximum in the accumulator space in case of images containing a circle, and we search for two maxima with significantly different radii, to increase robustness, in case of images containing an annulus. To reduce noise, the input image is smoothed by convolution with a Gaussian filter (5×5, $\sigma = 1$) for the Hough transform approach as well as for the following approaches.

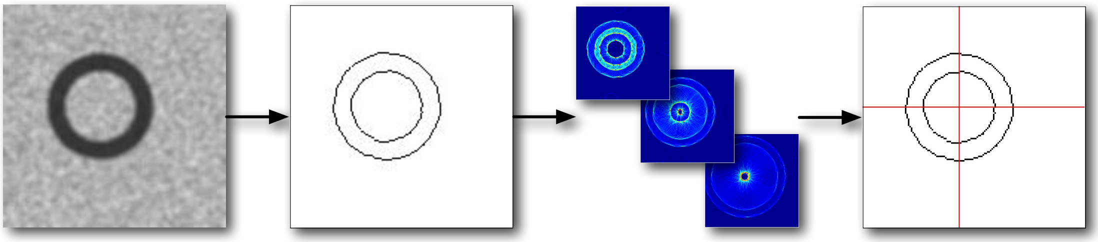


Figure 2.2.: Centre localisation based on the Hough transform. First, an edge detection algorithm is applied to the smoothed input image to obtain contour points; then, based on these contour points the accumulator array is computed; finally, the location of the maximum in the accumulator array is used as centre estimate.

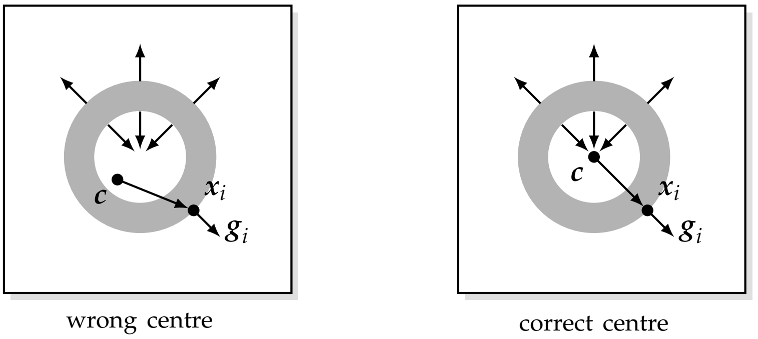
2.2.2. Gradient-based approach

Instead of using only the location of points on the object's contour, we can also use contour orientation; we therefore analyse the orientation of image gradients at positions of high variances of grey values, e.g. edges and corners. Figure 2.3 shows that the normalised displacement vector between a centre candidate $\boldsymbol{c}$ and a contour point $\boldsymbol{x}_i$ should have the same orientation (except for the sign) as the gradient $\boldsymbol{g}_i$ at $\boldsymbol{x}_i$ if $\boldsymbol{c}$ is the true centre. We quantify the connection between a centre, the displacement vectors, and the image gradient orientations by computing the dot product between the normalised displacement vectors $\boldsymbol{d}_i$ and the image gradients $\boldsymbol{g}_i$. The optimal centre $\boldsymbol{c}^*$ of a circular object in an image with N pixels at positions $\boldsymbol{x}_i$, $i \in \{1, ..., N\}$, is then given by

$$\boldsymbol{c}^* = \arg\max_{\boldsymbol{c}} J(\boldsymbol{c}) \ , \tag{2.1}$$

$$J(\boldsymbol{c}) = \frac{1}{N} \sum_{i=1}^{N} \left(\boldsymbol{d}_i^T \boldsymbol{g}_i \right)^2 \quad , \quad \boldsymbol{d}_i = \frac{\boldsymbol{x}_i - \boldsymbol{c}}{\|\boldsymbol{x}_i - \boldsymbol{c}\|_2} \ . \tag{2.2}$$

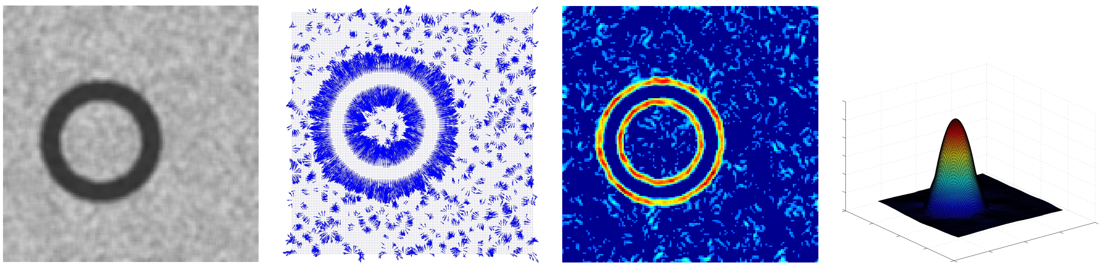
The displacement vectors $\boldsymbol{d}_i$ are scaled to unit length to be invariant to translations; the gradients $\boldsymbol{g}_i$ are scaled to unit length to account for strong highlights; furthermore, small image gradients can be ignored to increase efficiency. Figure 2.4 shows, exemplarily, the objective function $J(\boldsymbol{c})$; it yields a smooth function with a significant global maximum; farther from the global maximum, however, small local maxima may exist; hence, we must identify reasonable starting positions to guarantee convergence for an iterative scheme. Figure 2.4(c) indicates that positions with significantly large gradient magnitudes, i.e. contour points, can be used as starting positions.



wrong centre

correct centre

Figure 2.3.: Gradient-based approach for centre localisation. Left: The centre $\boldsymbol{c}$ is located such that the orientation of the displacement vector $(\boldsymbol{x}_i - \boldsymbol{c})$ differ from the absolute orientation of the gradient vector $\boldsymbol{g}_i$ at position $\boldsymbol{x}_i$; thus, the dot product between the displacement vector and its gradient vector $\boldsymbol{g}_i$ is large only for few positions $\boldsymbol{x}_i$. Right: The centre is located correctly and the sum of dot products reaches its maximum.



(a) input image (b) image gradients (c) gradient magnitude (d) objective function $J(\boldsymbol{c})$

Figure 2.4.: The low-pass filtered input image containing one annulus (a), its image gradients plotted as vectors (b), the gradient magnitudes where large values are red/white and small values are blue/black (c), and the corresponding objective function in the xy-plane (d).

Instead of evaluating J for several centres $\boldsymbol{c}$, we propose a gradient ascent approach for determining the maximum. The derivatives with respect to $\boldsymbol{c} = (c_1, c_2)^T$ are

$$\frac{\partial J}{\partial c_k} = \frac{2}{N} \sum_{i=1}^{N} \frac{(x_{ik} - c_k)\, e_i^2 - g_{ik}\, e_i\, n_i^2}{n_i^4} \quad , \tag{2.3}$$

where

$$n_i = \|\boldsymbol{x}_i - \boldsymbol{c}\|_2 \ , \quad e_i = (\boldsymbol{x}_i - \boldsymbol{c})^T \boldsymbol{g}_i, \quad \boldsymbol{g}_i = (g_{i1}, g_{i2})^T \ , \quad \boldsymbol{x}_i = (x_{i1}, x_{i2})^T \quad . \tag{2.4}$$

The iterative scheme for centre localisation based on image gradients is summarised in Algorithm 2.1; within each iteration step we compute a stepsize s using Armijo's rule [77], which is a common strategy for computing an efficient step size. Since the area around the global maximum of the objective function (see Figure 2.4(d)) can be narrow and the position of the largest magnitude can be outside the convergence area, especially for noisy images, we apply the outer iteration loop m times and determine the centre with the largest objective value as the optimal centre. Alternatively, we can also create a centre map of all detected centres, we can smooth this centre map to increase robustness, and we can determine the maximum of this smoothed centre map as the optimal centre; however, we then have to introduce new parameters such as the size of the bins or the parameters of the smoothing filter.

Figure 2.5 shows that the algorithm is fast and yields accurate results even for noisy and low contrast images. The accuracy and performance of our algorithm can be controlled by changing the number of trials m and the maximum number of iterations $t_{\max}$ for each trial. We can further improve performance by reducing the number of image gradients, for example, by ignoring image gradients with magnitude below a (predefined) threshold, e.g. the mean magnitude over all image gradients. We simply compute partial derivatives of the low-pass filtered image I by $\boldsymbol{g} = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})^T$ to obtain image gradients.

```
$c$ = centreLocalisation($\mathcal{G}, \mathcal{X}, m, t_{\max}, W, H$)

Input:   $\mathcal{G}$        image gradients $g_i$, $i = \{1, \dots, N\}$
         $\mathcal{X}$        position of image gradients $x_i$, $i = \{1, \dots, N\}$
         $m$        number of trials
         $t_{\max}$     maximum number of iterations
         $W, H$     image width and height
Output:  $c$        centre

$\theta \leftarrow 10^{-2}$                                        ▷ threshold to check convergence
$\mathcal{C} \leftarrow \{\}$                                          ▷ initialise set of detected centres
for $i \leftarrow 1, \dots, m$ do
    $c \leftarrow$ getInitialCentre($i$)                   ▷ position of the $i$th largest magnitude
    for $j \leftarrow 1, \dots, t_{\max}$ do
        $c_{\text{old}} \leftarrow c$                                            ▷ remember centre
        $g \leftarrow$ computeGradient($c, \mathcal{G}, \mathcal{X}$)         ▷ compute gradient according to Eq. 2.3
        $s \leftarrow$ computeStepsize($c, \mathcal{G}, \mathcal{X}, g$)   ▷ stepsize, computed via Armijo's rule [77]
        $c \leftarrow c + s\,g$                                    ▷ perform gradient ascent step
        if bordersReached($c, W, H$) then
            break                                   ▷ stop, if diverged to image borders
        end if
        if $\|c - c_{\text{old}}\| \leq \theta$ then
            $j \leftarrow$ computeObjective($c, \mathcal{G}, \mathcal{X}$)   ▷ compute value of objective function 2.2
            $\mathcal{C} \leftarrow (c, j)$            ▷ add detected centre and its objective value to the set
            break                                                ▷ stop, if converged
        end if
    end for
end for
$c \leftarrow \arg\max_{(c,j)\in\mathcal{C}} j$              ▷ determine detected centre with maximum value
```

Algorithm 2.1 (centreLocalisation): Acurate centre localisation based on the orientation of image gradients. To reduce computation time, $\mathcal{G}$ and $\mathcal{X}$ should only contain image gradients with large magnitudes, e.g. larger than mean gradient magnitude.

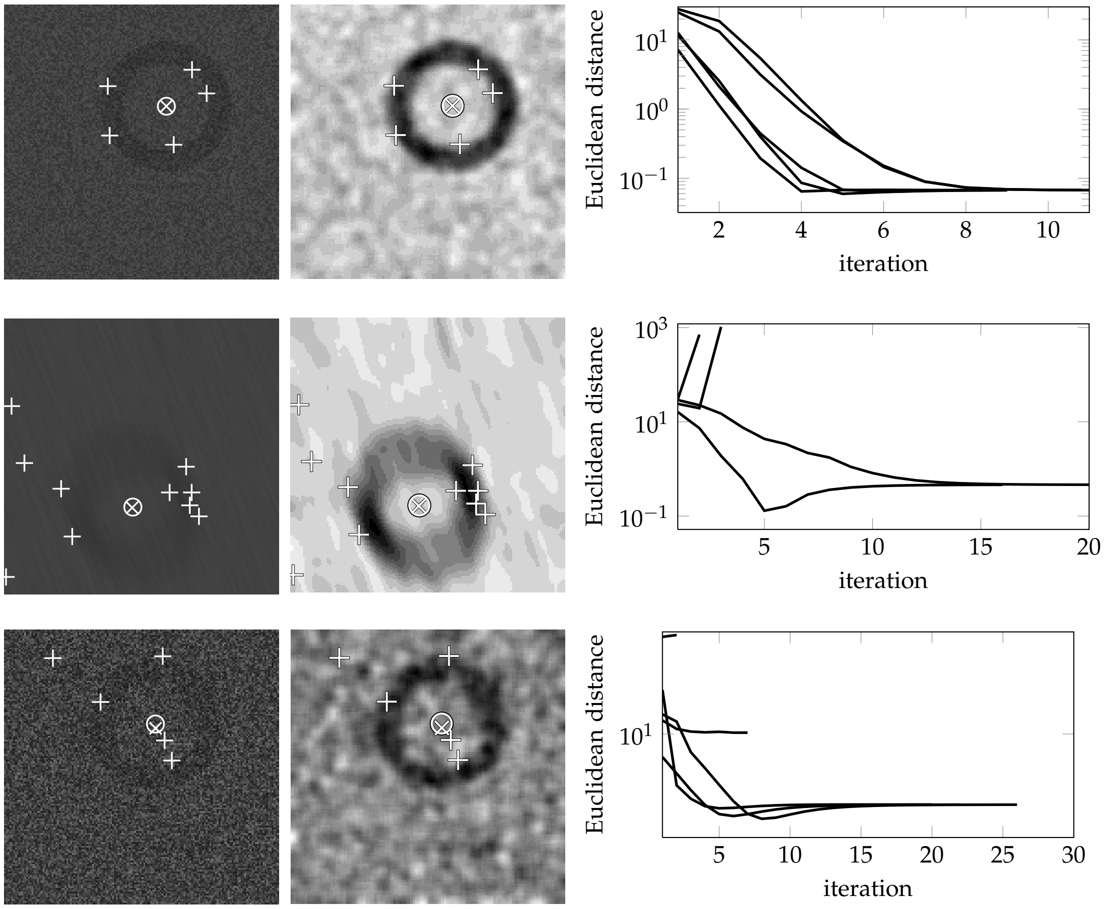


Figure 2.5.: Application of Algorithm 2.1 to low-contrast images (150 × 150 px) with different types of distortions (first row: white noise, second row: motion blur, third row: speckle noise). The detected centres are shown by crosses, the correct centres by circles, and starting positions for the incremental algorithm by plus symbols. The first column shows low-contrast input images, the second column shows the input images after smoothing (with a Gaussian filter of size 9 × 9 and with $\sigma = 3$) and stretched to full range, and the third column shows the error (Euclidean distance) between a current centre estimate and the true centre. Top row: In case of white noise, the structure of the ring and its image gradients are well-preserved and with $m = 5$ trials all starting positions converge to the same centre, which is very close to the optimum (error < 0.1 px); convergence is very fast, with 6 iterations on average. Second row: Even if the ring structure and the image gradients are partially distorted, e.g. by motion blur, the estimation is accurate (error < 1 px); although few starting positions diverged to the image borders, some converged to an almost optimal estimation; again, convergence is fast with 15 iterations on average. Last row: In case of speckle noise, the image gradients are locally distorted such that the centre estimation is not as accurate as for the former images; however, the error is less than 2 px and most of the starting positions converged to the same solution; once more, convergence is fast with 15 iterations on average.

2.2.3. Symmetry-based approach

In case of higher image degradations, e.g. low contrast in combination with strong noise, gradient information is insufficient for estimating the centre accurately. We, therefore, propose a second approach for centre estimation, which is based on the minimisation of radial intensity variations.

Assume we have an annulus with only few colour variations inside, then there are several isophotes, i.e. contours of equal intensity, sharing the same centre. Thus, the mean radial colour variation reaches its minimum for the correct centre. Based on this observation, we define the notion of radial symmetry R for a particular centre $\boldsymbol{c}$ in the image I by

$$R(\boldsymbol{c}) \quad = \quad \frac{1}{L}\sum_{j=1}^{L}\sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(I^*_{\boldsymbol{c}}(i,j)-\mu_j\right)^2}\ , \tag{2.5}$$

with

$$I^*_{\boldsymbol{c}}(y,x) \quad = \quad I\left(\operatorname{atan}\left(\frac{y-c_y}{x-c_x}\right), \|\boldsymbol{x}-\boldsymbol{c}\|_2\right)\ , \quad \text{and} \tag{2.6}$$

$$\mu_j \quad = \quad \frac{1}{M}\sum_{i=1}^{M} I^*(i,j)\ , \tag{2.7}$$

where I^* is the polar transform of image I with the origin at $\boldsymbol{c}$, M is the number of *slices*, L is the number of samples on each slice, and μ_j is the mean value of distance j over all slices/orientations. Figure 2.6 shows the definition of a slice s_k, which captures intensities along a particular orientation and with respect to a particular centre $\boldsymbol{c}$.

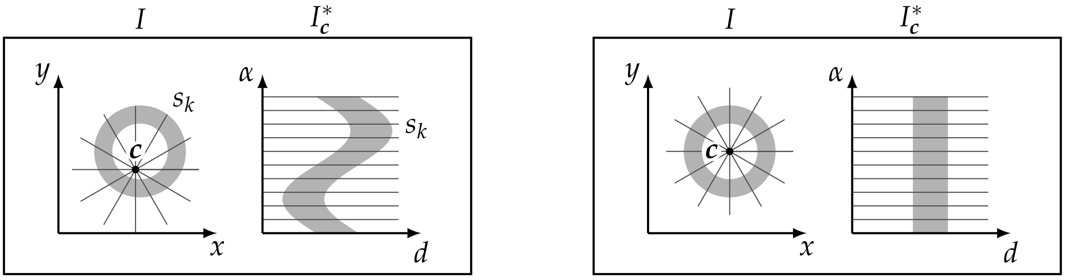
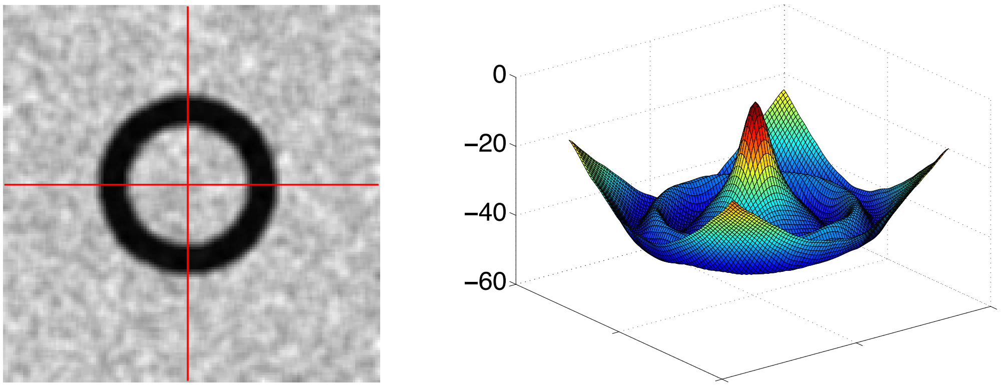


Figure 2.6.: Transformation 2.6 applied to an example image I. The accuracy of the transformation is determined by the number of equally spaced samples, with a distance d from the current centre, on each slice and the number of slices/angles α. Left: Due to high variations of the intensities in direction α for given d, $R(\boldsymbol{c})$ is large for incorrect centres. Right: The transformed annulus appears as a strip with no vertical intensity variations; hence, $R(\boldsymbol{c})$ reaches its minimum for the correct centre.



(a) low-pass filtered input image

(b) objective function $R(\boldsymbol{c} = (x, y))$ evaluated in the (x, y) image space

Figure 2.7.: Left: Example image and the centre that has been detected by minimising radial intensity variations. Right: The inverted objective function $R(\boldsymbol{c})$ in the xy-plane shows a significant maximum.

Then, the optimal centre $\boldsymbol{c}^*$ is given by

$$\boldsymbol{c}^* = \arg\min_{\boldsymbol{c}} R(\boldsymbol{c}) \ . \tag{2.8}$$

Since an iterative scheme for solving Problem 2.8 cannot be derived in closed-form, we evaluate R on a predefined grid; the grid position with minimum R, then, yields the estimated centre. The accuracy of the centre estimation is determined by the resolution of the grid on which R is evaluated; however, the computation time grows quadratically with the resolution of the grid. If not otherwise noted, we compute R on the same grid that is used for the Hough transform, i.e. a resolution of 0.5 on both axis, to ensure a fair comparison. Figure 2.7 shows an example of the objective function and the estimated centre.

2.2.4. Results With Synthetic Images

We evaluate the performance of the three approaches on a synthetic dataset with images of $150 \times 150\,\text{px}$ containing a randomly centred annulus or circle of fixed size; the grey values of the annulus/circle were uniformly distributed within the interval $[40, 70]$ and the grey values of the background within $[50, 80]$. Figure 2.8 shows that the images contain white noise and that the contrast is very limited. We further added multiplicative noise (speckle) and motion blur; in total, our synthetic dataset contains 600 images, 100 images for each type of noise and for each object (annulus/circle). We compare the performance of the different approaches by computing the mean Euclidean

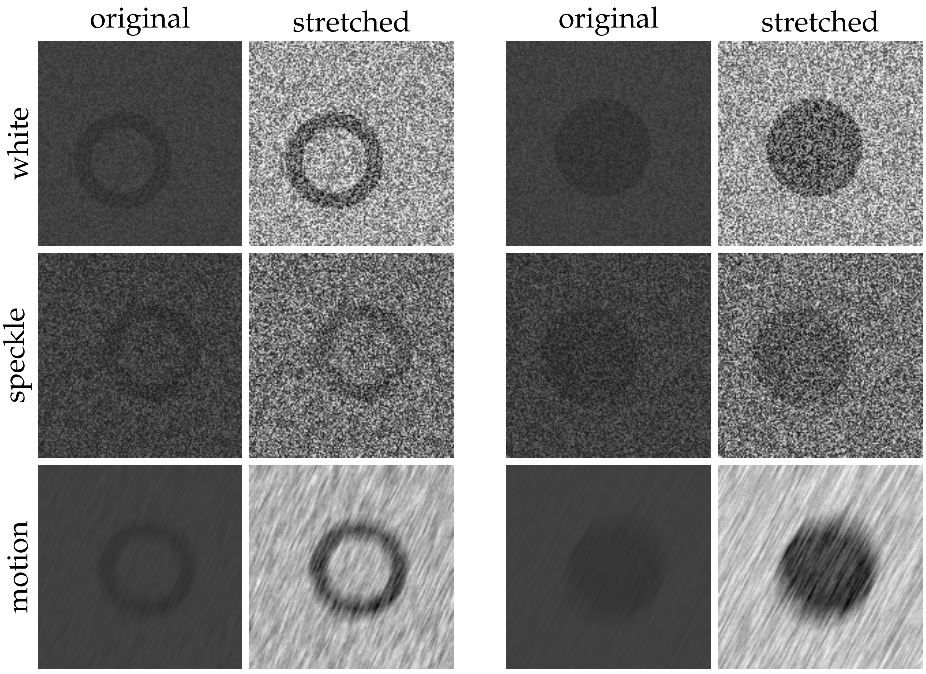


Figure 2.8.: Example synthetic images containing an annulus or a circle with different types of distortions: white noise, speckle noise, and motion blur; for comparison, the images are stretched to full range.

distance between the correct centre and the estimated centre as well as the standard deviation.

We apply the gradient-based approach with m =50 trials and $t_{\max} = 100$ iterations per trial to obtain the centre estimate with maximum value. Even though the values of m and $t_{\max}$ we have used previously, compare Figure 2.5, were significantly smaller, we, here, use a larger number of trials and iterations to increase robustness. Note that the maximum number of iterations is only reached in few cases, since we stop the iterative maximisation if the position does not change significantly.

Since the variance-based approach requires the evaluation of several centre candidates, we evaluate the objective function $R(\boldsymbol{c})$ on the same grid that is used for the Hough transform approach, i.e. a resolution of 0.5 on both axis; finally, we identify the position with minimum value. The intensity values are radially analysed with 100 slices s_k; each slice has a length of 75 px, which is half the image size, and contains 100 equally spaced positions; the intensity for these positions is computed by bilinear interpolation.

Table 2.1 shows the results for the 600 synthetic images. In general, the performance on images containing an annulus is superior compared to the performance on images containing a circle; an annulus has two contours and, therefore, shows almost twice

Table 2.1.: Performance comparison for 6 datasets, each consisting of 100 artificially created images with low-contrast and different types of noise. The mean error (in pixel) is evaluated over the 100 images for each method; standard deviations are depicted in parentheses. The symmetry-based approach significantly outperforms the other approaches with a minimum error of 0.07 px for images containing an annulus and with white noise; even the maximum error of the symmetry-based approach for images with a circle and speckle noise is below 1 px, which demonstrates its robustness, especially, for strong image degradations.

(a) white noise

method	circle	annulus
Hough transform	0.91 (0.55)	0.79 (0.62)
gradient-based	1.01 (0.63)	0.78 (0.49)
symmetry-based	**0.71 (0.11)**	**0.07 (0.16)**

(b) speckle noise

circle	annulus
9.89 (13.74)	9.39 (13.38)
8.70 (10.21)	4.64 (2.65)
0.89 (0.71)	**0.59 (0.55)**

(c) motion blur

method	circle	annulus
Hough transform	17.59 (16.44)	14.90 (14.54)
gradient-based	2.55 (1.66)	1.64 (1.35)
symmetry-based	**0.52 (0.59)**	**0.21 (0.42)**

as many gradients as a circle; consequently, accuracy improves for our two novel approaches as well as for the Hough transform approach in case of annulus images.

In case of images with white noise the Hough transform and the gradient-based approach yield comparable results with a mean error rate of approximately 1 px for images containing a circle and a mean error rate of 0.79 px for images containing an annulus. The approach based on radial symmetry significantly outperforms the other approaches with an mean error rate of 0.71 px (circle) and 0.07 px (annulus).

In case of images with speckle noise the performance of all approaches decreases, but the symmetry-based approach yields significantly better results with a mean error of still below one pixel. The Hough transform and the gradient approach perform comparable for circles (error: 8–10 px). In case of annulus images the gradient approach clearly outperforms the Hough transform, since the number of large image gradients is almost twice compared to images containing a circle; hence, the gradient approach is more robust to speckle noise.

In case of images with motion blur the object's shape and contour partially change such that the Hough transform cannot detect the correct centre accurately (mean error for circles: 17.59 px, for annuli: 14.90 px); in constrast, the gradient and the symmetry approach yields accurate results. Since the symmetry-based approach can compensate for partial distortions of the contour, it outperforms significantly the other two methods (error of 0.52 px for circles and 0.21 px for annuli).

In total, the radial symmetry approach achieves the best overall performance for noisy and low contrast images with an error of less than 1 px on average, and the gradient-based approach outperforms the Hough transform in all cases of higher image degradations.

Concerning the computation time, the two novel approaches yield superior performance compared to the Hough transform as they reduced the computation time by a factor of 2 (symmetry-based approach) and by a factor 8 (gradient-based approach).

2.2.5. Discussion

We have shown that the centre of a circular object can accurately be detected for low-contrast and noisy images. Even if the shape of the object is corrupted, we can determine the object's centre with an error of less than 1 px. We used two novel approaches for the centre localisation; the first approach is based on the orientation of image gradients with a novel objective function, which is maximised by a simple and fast gradient ascent technique; the second approach is based on the observation that radial intensities of a circular object show small variations for the correct centre and large variation for an incorrect centre.

Compared to existing methods, the gradient-based method directly incorporates the potential centre into the objective function and leads to a very simple cost-function, which is based on dot products only; hence, we avoid an exhaustive search and compute the centre efficiently. If, however, the image gradients of the object's contour are extensively distorted and its shape changes entirely, the performance of the gradient-based method decreases; in these cases, the symmetry-based method should be applied, since it does not employ image gradients.

We created synthetic, low contrast images with a randomly centred annulus or circle of fixed size. Furthermore, we added three different types of noise: white noise, speckle noise, and motion blur. We extensively compared the accuracy of our approaches with an approach based on the Hough transform.

The radial symmetry approach achieved the best overall performance (mean error: 0.49 px) and the gradient-based approach outperforms the Hough transform in all cases of higher image degradations. Our novel approaches reduce the computation time considerably compared the Hough transform approach, and we can easily control computation time by changing the parameters, e.g. the accuracy of the slices or the number of iterations, such that both approaches can be applied to a wide range of applications such as eye tracking or object tracking in real-time. In industrial applications where the centre must be estimated precisely, e.g. calibration or inspection, the variance approach can provide the most accurate centre estimations. We will demonstrate the performance of the novel approaches for real-world datasets in Part II, where we successfully use them for the inspection of LEDs and for accurate eye centre localisation.

The disadvantage of our variance-based approach is that it requires the evaluation of several centre candidates and an automatic preselection of reasonable centre candidates can be difficult and may lead to inaccurate estimations. It is a question for future research to investigate whether the optimal solution can be obtained more efficiently, for example, by approximating the partial derivatives to derive an iterative scheme. Moreover, we can also use a combination of both approaches, e.g. the gradient-based approach is applied to select centre candidates that are used as input for the variance-based approach.

2.3. Illumination Correction

Digital images suffer from various deficiencies of the hardware that was used for image acquisition, including sensor, lenses, and, especially, illumination. Even with a perfect sensor, a lens without aberrations, and a perfectly homogeneous illumination, the image may still show an intensity falloff towards the corners of the image due to natural vignetting. In general, the illumination inhomogeneity is much more complex [4] and cannot be described with a simple model.

In various applications, such as 3D-texture mapping, aerial photography, or astronomy, illumination gradients must be removed when creating image mosaics or registering image sequences. Without correction, artefacts may clearly be visible and mislead further image processing or pattern recognition systems. In the case of surface inspection, for example, small and subtle defects must be detected and inhomogeneities can lead to a large number of missed defects.

The problem of illumination correction has been addressed by many authors, mostly with respect to a particular illumination artefact; vignetting correction methods have been proposed in [119] and [55], for example; in magnetic resonance imaging, a method based on information minimisation has been proposed, where the correction components are modelled as combinations of smoothly varying basis functions [68]; in face recognition, methods for illumination compensation have been proposed as a preprocessing step [2, 66], primarily to find a more invariant face representation; more recently, a non-parametric shading correction method has been proposed in [85]. An overview of stitching and blending methods can be found in [65], for example.

We aim for an illumination correction method that removes inhomogeneities accurately and that qualifies for real-time applications such as virtual material design. Hence, we focus on a simple method based on lowpass filtering using Gaussian pyramids and appropriate image extrapolation to avoid boundary artefacts.

2.3.1. The Gaussian Pyramid

For more than two decades, pyramid methods have been used for image processing tasks such as image enhancement, compression, interpolation and extrapolation of missing image data, and numerous others [78].

Given a grey-level input image with intensity values $G_0(x, y)$ at discrete locations $x \in [0, m-1]$ and $y \in [0, n-1]$ the Gaussian pyramid is an efficient data structure for spectral decomposition. For each level i the image $G_i(x, y)$ is lowpass filtered and downsampled by a factor of two to produce the image $G_{i+1}(x, y)$, which is commonly referred to as the `reduce` operation.

$$\begin{aligned} G_{i+1} = \texttt{reduce}(G_i) &= (\downarrow 2)(h * G_i) \quad \text{with} \\ (\downarrow 2) f(x, y) &= f(2x, 2y)\,. \end{aligned}$$

This is done successively up to a certain level l. Then, the `expand` operation is applied successively to get an image with the same size as G_0, but containing only the frequency components of G_l:

$$G'_{i-1} = \texttt{expand}(G_i) = 4\,h * ((\uparrow 2) G_i)\,.$$

The weighting function h is often called the *generating kernel* and is commonly chosen to be a 5-by-5 binomial filter to approximate the Gaussian. This method for computing the low frequency components of the input image is obviously more efficient than direct convolution with a large filter but also more efficient than using standard FFT [1].

Finally, we need to decide how the image data is extended when filtering boundary pixels; this is the critical step where boundary artefacts occur. In the following, we present a general framework that covers not only the extrapolation conditions for rectangular images, but also arbitrary image shapes that occur, for instance, during image segmentation.

2.3.2. Boundary Extrapolation

Since the filtering operation is defined as

$$(f * h)(x, y) = \sum_{i=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} h(i, j) f(x-i, y-j)$$

the image must be extended by two pixels—either virtually or explicitly—in each direction when using 5-by-5 filters. Commonly, pixels outside the image are set constant or to the value of the nearest boundary pixel (replicate boundary), or the pixels are computed by assuming a periodic image (circular boundary).

Given the input image $f(x, y)$ with $x \in [0, m-1]$ and $y \in [0, n-1]$ we seek an extended image $f'(x,y)$ with $x \in [-2, m+1]$ and $y \in [-2, n+1]$. Next, we define for each pixel (x,y) a set of pixels

$$\mathcal{P}_{x,y} = \left\{ \left(p_i^x, p_i^y\right) \right\}_{i=1}^{|\mathcal{P}_{x,y}|}$$

that influence the intensity at (x,y) in the extended image. Finally, the function $g_{\mathcal{P}_{x,y}}(x,y)$ defines how to calculate the intensity at (x,y) from the intensities of the set of pixels $\mathcal{P}_{x,y}$. Thus, we get

$$f'(x,y) = \begin{cases} f(x,y) & \text{if } x \in [0, m-1] \text{ and } y \in [0, n-1] \\ g_{\mathcal{P}_{x,y}}(x,y) & \text{otherwise.} \end{cases} \tag{2.9}$$

To simplify notation, we use $g(x,y)$ instead of $g_{\mathcal{P}_{x,y}}(x,y)$. Next, we show how the standard boundary conditions, replicate and circular, as well as different extrapolation methods fit into this framework.

Boundary Replication

Assuming a replicate boundary, the intensity values of pixels outside the image equal those of the nearest boundary pixel:

$$\mathcal{P}_{x,y}^{\text{rep}} = \left\{ \left(p_0^x, p_0^y\right) \;\middle|\; \begin{array}{l} p_0^x = \min(\max(x,0), m-1) \\ p_0^y = \min(\max(y,0), n-1) \end{array} \right\}.$$

Thus, $\left|\mathcal{P}_{x,y}^{\text{rep}}\right| = 1$ for all x and y, and $g^{\text{rep}}(x,y) = f(p_0^x, p_0^y)$. Analogously, a circular boundary condition can be defined:

$$\mathcal{P}_{x,y}^{\text{circ}} = \left\{ \left(p_0^x, p_0^y\right) \;\middle|\; \begin{array}{l} p_0^x = x \bmod m \\ p_0^y = y \bmod n \end{array} \right\}.$$

Again, $|\mathcal{P}_{x,y}^{\text{circ}}| = 1$ for all x and y and $g^{\text{circ}}(x,y) = f(p_0^x, p_0^y)$.

Linear Extrapolation

Both replicate and circular boundary assumptions are invalid assumptions when modelling illumination gradients. In almost all cases of natural illumination inhomogeneity the intensity gradient continues smoothly outside of the image boundary; hence, replicate or circular boundary extrapolation overestimates the intensity gradient at the boundary, when compensating for vignetting.

A first step towards more realistic boundary assumptions involves simple linear extrapolation of the intensity values at the boundary. The gradient is the difference between a boundary pixel and its nearest neighbour towards the centre of the image; mathematically, we can express this by:

$$\begin{aligned}
\mathcal{P}^{\text{lin}}_{x,y} &= \left\{ \left(p^x_0, p^y_0\right), \left(p^x_1, p^y_1\right) \right\} \quad \text{with} \\
p^x_0 &= \min(\max(x,0), m-1) \\
p^y_0 &= \min(\max(y,0), n-1) \\
p^x_1 &= \min(\max(x,1), m-2) \\
p^y_1 &= \min(\max(y,1), n-2) \quad \text{and} \\
g^{\text{lin}}(x,y) &= f\left(p^x_0, p^y_0\right) + \left(f\left(p^x_0, p^y_0\right) - f\left(p^x_1, p^y_1\right)\right) \cdot \left\| \begin{pmatrix} x - p^x_0 \\ y - p^y_0 \end{pmatrix} \right\|_1 ,
\end{aligned}$$

where $\| \cdot \|_1$ denotes the Manhattan distance. This modification improves the illumination compensation significantly and can be extended, for instance, to least-squares regression with two-dimensional second order polynomials as shown below.

Least Squares Regression

We define the set $\mathcal{P}_{x,y} = \left\{ \left(p^x_i, p^y_i\right) \right\}$ to contain a 5-by-5 block of pixels within the image bounds that minimises the mean distance to the point (x, y). Then, we can formulate the following minimisation problem

$$\begin{aligned}
\min_\beta \quad & \|\mathbf{A}\beta - \mathbf{b}\|_2 \quad \text{with} \\
\mathbf{A} = & \begin{pmatrix} p^x_0 & p^y_0 & (p^x_0)^2 & (p^y_0)^2 & p^x_0 p^y_0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p^x_{24} & p^y_{24} & (p^x_{24})^2 & (p^y_{24})^2 & p^x_{24} p^y_{24} & 1 \end{pmatrix} \\
\mathbf{b} = & \begin{pmatrix} f(p^x_0, p^y_0) \\ \vdots \\ f(p^x_{24}, p^y_{24}) \end{pmatrix} .
\end{aligned}$$

The solution is computed by

$$\beta = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1} \mathbf{A}^{\mathrm{T}}\mathbf{b} \ ,$$

and the intensity at (x, y) is approximated by

$$g(x,y) = \begin{pmatrix} x & y & x^2 & y^2 & xy & 1 \end{pmatrix} \cdot \beta \, .$$

Note that other regression approaches may also be used. However, the complexity of a regression model, i.e. the number of parameters to be estimated, should be low, as illumination gradients are by definition smooth. The choice of the neighbourhood size is a trade-off between stability and runtime, but 5-by-5 blocks are sufficient to remove almost all visible boundary artefacts as demonstrated by our experiments. The computational complexity of the regression method is no crucial factor, as extrapolation operates on few boundary pixels only; other operations such as image filtering dominate the complexity of the illumination correction method.

Arbitrary Boundary Shapes

In some cases, illumination gradients must be removed from images that contain more than one texture. Here, the different textures should be segmented first and individually corrected afterwards. In general, these regions are non-rectangular and so the above methods must be adapted for arbitrary shaped images. Now, the input image $f(x,y)$ is defined only in certain regions, i.e. the image contains valid intensities for a set of pixels $I \subset [0, m-1] \times [n-1]$. Then, Equation 2.9 is modified slightly such that

$$f'(x,y) = \begin{cases} f(x,y) & \text{if } (x,y) \in I \\ g_{\mathcal{P}_{x,y}}(x,y) & \text{otherwise.} \end{cases} \tag{2.10}$$

In this more general case, it is not feasible to pick $\mathcal{P}_{x,y}$ from a fixed-size neighbourhood of a certain shape, e.g. when I is very sparse or has a fractal-like boundary; instead $\mathcal{P}_{x,y}$ should contain a fixed number of adjacent pixels from within I.

2.3.3. Algorithm

So far, we focused on the boundary extrapolation during filtering, but the aim is to correct illumination inhomogeneities in texture images. In the overall algorithm (see Algorithm 2.2), extrapolation of boundary pixels is performed by the function `borderExtrapolation`.

In the downsampling loop, the image is first extended by two rows and columns in each direction and then the intensity values of these pixels are determined by extrapolation; after filtering, the image is cropped to the original size. In the upsampling loop, the image is first filtered and then the intensity values of the first and last two rows and columns are recalculated from the inner image by extrapolation. Finally, the low-frequency image is subtracted from the original image and the mean intensity of the input image is added to obtain the final image with proper intensity levels.

2.3.4. Experiments

We demonstrate the performance of our method in a series of experiments, for which we used grey-level images containing intensity values in the range $[0, 255]$; to avoid discretisation artefacts we performed each operation in double-precision arithmetic.

First, we used artificial images that contained no texture but only smooth intensity gradients as they occur with natural vignetting (see Figure 2.9). Perfect illumination correction would yield a completely homogeneous image; obviously, the choice of the boundary condition does not affect the centre of the image, which shows a slight gradient even after correction. At the image boundary the replicate condition is outperformed by linear and polynomial extrapolation, the latter yielding an difference image intensity range of less than 1, which is optimal in case of 8 bit images.

Second, we measured how boundary artefacts affect the histogram of the intensity values. Therefore, we calculated the difference between the histograms of the image boundary (64 pixels wide) and the image centre (see Figure 2.10). The replicate condition causes the corrected image to be too dark at the boundary, i.e. the histogram of the

```
$I^* = \texttt{illuminationCorrection}(I, l)$
Input:   $I$     input image of size $W, H$
         $l$     pyramid depth
Output:  $I^*$   corrected image

$G_0 \leftarrow I$                                              ▷ initialise
for $i \leftarrow 0, \dots, l-1$ do                             ▷ pyramid down
    $G_i \leftarrow \texttt{extend}(G_i)$                         ▷ extend image
    $G_i \leftarrow \texttt{borderExtrapolation}(G_i)$            ▷ extrapolate borders
    $G_i \leftarrow h * G_i$                                    ▷ image filtering
    $G_i \leftarrow \texttt{crop}(G_i)$                           ▷ crop to original size of $G_i$
    $G_{i+1} \leftarrow (\downarrow 2)(G_i)$                      ▷ downsampling
end for
$G_l^* \leftarrow G_l$                                          ▷ initialise
for $i \leftarrow l, l-1, \dots, 1$ do                          ▷ pyramid up
    $G_{i-1}^* \leftarrow (\uparrow 2) G_i^*$                     ▷ upsampling
    $G_{i-1}^* \leftarrow h * G_{i-1}^*$                          ▷ image filtering
    $G_{i-1}^* \leftarrow \texttt{borderExtrapolation}(G_{i-1}^*)$  ▷ extrapolate borders
    $G_{i-1}^* \leftarrow 4\, G_{i-1}^*$                          ▷ correct energy
end for
$I^* \leftarrow I - G_0^* + \frac{1}{WH} \sum_{x,y} I(x,y)$     ▷ shift intensity of output image
```

Algorithm 2.2 (`illuminationCorrection`): General framework for illumination correction based on the Gaussian pyramid.

boundary pixels is shifted towards zero and the histogram difference has a maximum for small values. Whereas, with linear extrapolation this effect is dramatically reduced; polynomial extrapolation yields only a slight further improvement.

Figure 2.11 depicts some typical textures as they occur in virtual material design. The images were acquired with an industrial colour camera (1392 × 1040 pixels, 8bit); they show clearly visible intensity falloffs towards the image corners. We applied the above described method for each colour channel individually and compared it to the replicate boundary condition. The intensity gradients are invisible after correction, however, other inhomogeneities and artefacts are still visible such as repetitive structures (see rows 2 and 3) or very subtle blurring towards the image corners due to lens deficiencies.

2.3.5. Discussion

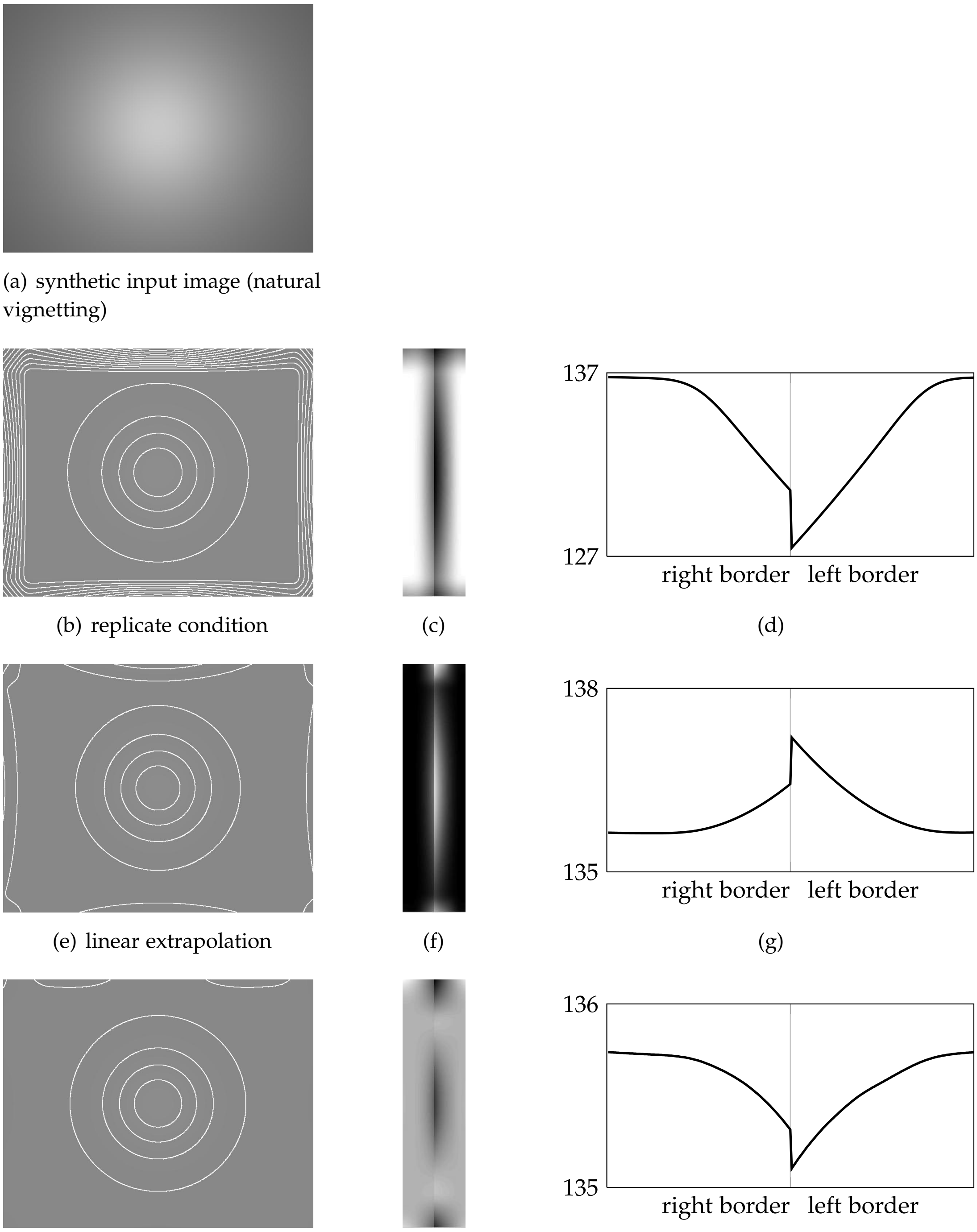
We have shown that removing illumination inhomogeneities from texture images using the Gaussian pyramid yields accurate results. Whereas with standard filtering using replicate or circular boundary condition the resulting images show artefacts at the image borders, a linear or a polynomial boundary condition results in almost homogeneous images with minimum border artefacts—in case of 8bit images border artefacts disappear completely.

We have introduced a framework by which inhomogeneous illumination and border artefacts are removed using different types of boundary extrapolation. The framework is not limited to linear or polynomial extrapolation, but experiments indicate that no improvement can be expected using more complex functions, since performance is already close to optimum when using simpler extrapolation.

The Gaussian pyramid provides a fast way to model arbitrary illumination gradients. All calculations can be done in real-time—the proposed boundary extrapolation techniques have low complexity, since an image has generally few boundary pixels. In case of colour images, we apply the illumination correction for each channel separately; we reach 20 frames per second for colour images (1280 × 1024 px, 24bit) and with standard computer hardware.

Especially in the case of inspecting texture images for subtle deviations, an image gradient can lead to many false positives. Since simple techniques for gradient removal still show artefacts at the image boundaries, we expect false positives in these regions even after correction. Therefore, we will apply the proposed illumination correction with linear extrapolation during preprocessing of texture images to remove inhomogeneities completely and to normalise illumination.

Finally, we discuss the role of illumination correction in texture synthesis as it is used in virtual material design. Besides intensity gradients, other image acquisition artefacts such as blurred image regions, reflexions, or saturation may occur. All of them will cause the results of naïve image stitching to look repetitive and unrealistic, especially



(a) synthetic input image (natural vignetting)

(b) replicate condition

(c)

(d)

(e) linear extrapolation

(f)

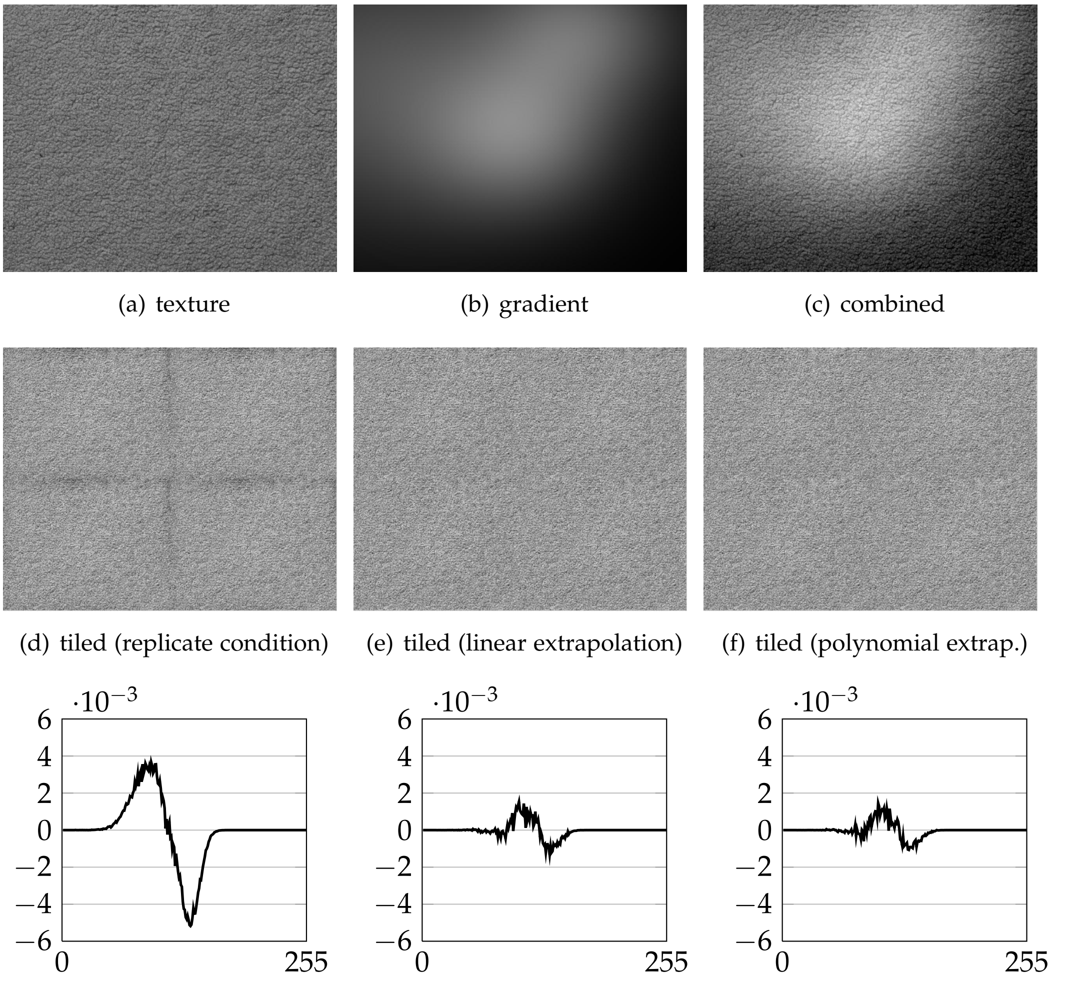
(g)

(h) polynomial extrapolation

(i)

(j)

Figure 2.9.: Comparison of illumination correction methods for an example image. The intensity levels of the corrected images (left column) are emphasised with contours. Additionally, an image transition (middle column, 128 pixels from each side, intensity stretched to full range) and the intensity across the middle row (right column) are shown. Note the different scales on the y-axis and that the boundary artefacts are reduced.



(a) texture (b) gradient (c) combined

(d) tiled (replicate condition) (e) tiled (linear extrapolation) (f) tiled (polynomial extrap.)

(g) border histogram (replicate) (h) border histogram (linear) (i) border histogram (polynomial)

Figure 2.10.: Boundary artefacts in image mosaics. The top row shows a texture (a), a simulated illumination gradient (b) and the combination of both (c). The middle row shows the correction results for three methods as 2-by-2 image mosaics. The replicate condition (d) produces dark areas at image transitions; with linear (e) or polynomial extrapolation (f) the transitions are almost invisible. The difference between the histograms of centre and boundary pixel intensities (bottom) verifies this observation.

at the image borders where textures are non-continuous. One solution would be to enhance the original image and remove as many artefacts as possible. However, a texture synthesis approach, such as proposed by Efros and Freeman [32], can compensate for the image acquisition artefacts by synthesising larger textures from smaller blocks of the original image. Each new block is chosen to optimise an overlapping region and, finally, the blockiness of the boundary is reduced by finding the minimum cost path within the overlapping region. In this scenario, illumination correction is, again, a fundamental preprocessing step to avoid intensity gradients and our method in connection with texture synthesis can yield natural-looking textures of arbitrary size and shape.

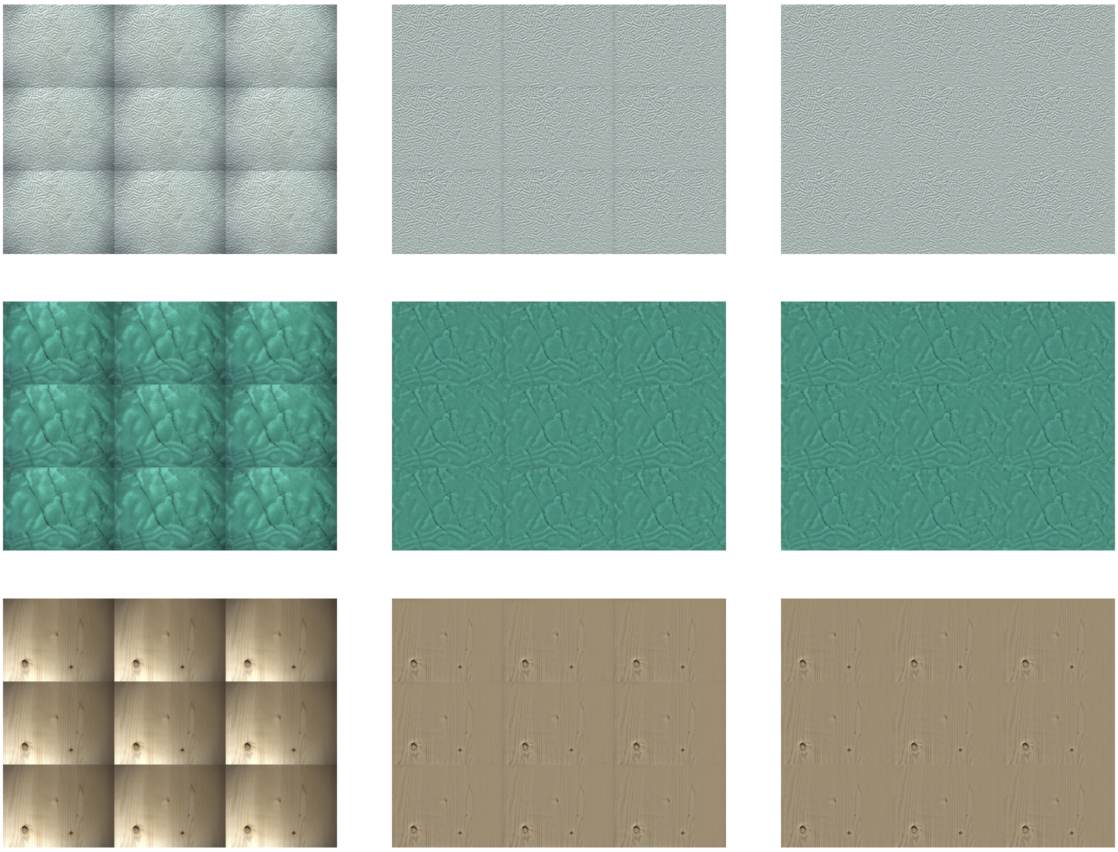
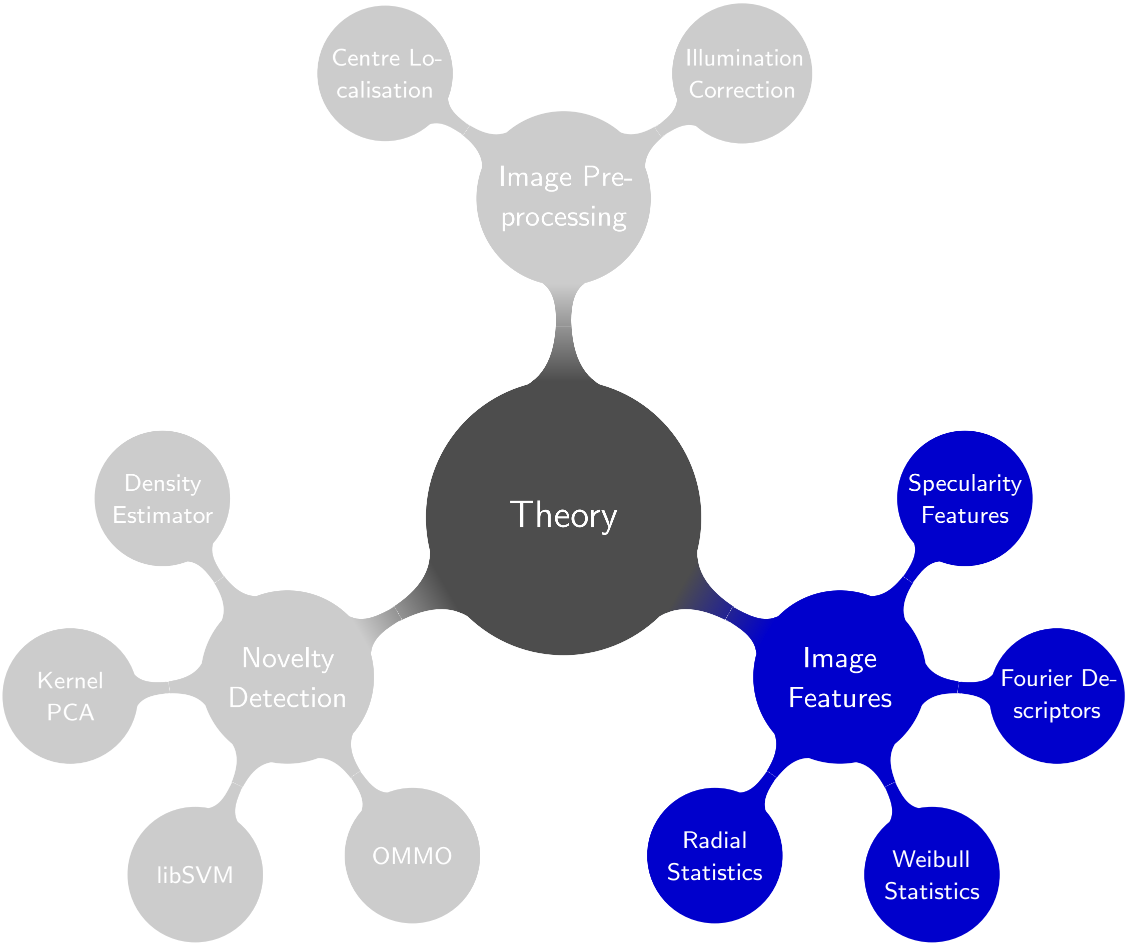


Figure 2.11.: Real-world colour textures. The performance of the illumination correction is shown for some real-world textured materials (from top to bottom: paper towel, green linoleum, wood) in 3-by-3 image mosaics. Without illumination correction (left column) vignetting is most obvious; with illumination correction using a replicate boundary condition (middle column) vignetting is still visible; with illumination correction using a polynomial boundary extrapolation (right column) vignetting is removed almost completely.

3. Image Features



Each feature extraction method proposed in this chapter has been developed completely by myself; this includes the idea, derivation, implementation, and validation. Some of the work described in this chapter has previously been published in [101, 105, 104, 103] or is currently under review [98]. Parts of the proposed methods have been integrated into industrial applications (see Part II).

3.1. Introduction

The computation of appropriate image features plays a crucial role in the development of a machine vision system, since they serve as input for later stages such as classification and novelty-detection. Even a theoretically optimal classifier will lead to poor performance, if the computed image features fail to accurately describe the characteristics of the objects to be classified.

Feature extraction has long been a large research field with various applications; in medical imaging, image features capture characteristics of tumour and healthy tissue; in multimedia, image features have been designed such that a motorcycle and a car can be distinguished and large image databases can be organised; in vision science, image features are used to describe the focus of attention in natural images and natural videos; in robot vision, image features describe properties of scene objects such as doors, stairways, desks, or chairs; in industrial imaging, image features are used to describe shape or appearance of semi-conductor components.

Since all these feature extraction methods must provide highest performance and highest reliability, they are specifically designed for a particular application, especially in case of industrial applications; hence, they achieve inferior results when applied across different areas or even across different applications within the same area.

In general, feature extraction methods can roughly be divided into geometrical, statistical, frequency-based, model-based, and hybrid approaches; we can further categorise them as local or global approaches or as approaches that work on different image scales, for example. A survey of various methods for feature extraction and their categorisation can be found in [87], for example.

In this chapter, we introduce four novel feature extraction methods that capture characteristic image properties of defective and defect-free samples; we only briefly motivate and derive the approaches here, since we extensively evaluate and compare our methods for real-world applications in Part II.

First, we present a hybrid approach that computes statistics of geometrical features and of shape features; this approach can capture, for instance, the specularity of surfaces. Geometrical features are mostly computed based on a single binary image, however, if the image contains objects at different grey-levels the geometrical features must be computed across grey-levels. Then, the question is how these geometrical features are combined to yield a proper description of the image characteristics. We, therefore, use a general decomposition framework that separates the input image into a stack of binary images at different grey-levels; then, connected components are determined and analysed for certain properties such as orientation, compactness, or eccentricity; features from each component are averaged first and combined by first order statistics for each binary image as well as across all binary images afterwards.

Second, we use the decomposition framework and evaluate the characteristics of components in a single binary image by using Fourier descriptors. So far, Fourier descriptors have either been used for a single binary image or, more recently, for a single binary image at different image scales [63]. We demonstrate that our general decomposition framework in combination with Fourier descriptors can capture characteristic image features that occur due to specular reflections.

Third, we present a novel approach for the analysis of radial intensity variations; this approach is closely related to the centre detection approach presented in Section 2.2.3 and is motivated biologically. With our novel approach we can capture small and subtle defects, even in the presence of strong image noise.

Fourth, we propose an approach that analyses local image gradients; the idea is that the distribution of image gradients in defective regions will significantly deviate from that of image gradients in defect-free regions. We, therefore, compute a Weibull fit to the distribution of local image gradients and use the estimated parameters of the Weibull distribution for novelty detection. Recently, experiments have shown that brain responses strongly correlate with image statistics described by Weibull features when viewing natural images [91]. So far, Weibull image features have only been applied to classify images globally and to organise image databases [116]. We demonstrate that Weibull features can be applied to capture local variations as well, for instance, of texture images.

3.2. Specularity Features

The characterisation and inspection of specular surfaces[1] is used in a wide range of industrial applications such as solder joint inspection or welding seam inspection. Theoretically, the inspection of these surfaces can be simplified if optimal hardware is used such as a telecentric lens and a telecentric illumination or 3D techniques. However, such hardware is expensive and hard to integrate into an existing manufacturing process; with standard hardware the surface cannot be modelled in 3D as precisely as required to detect subtle defects with high reliability. Therefore, feature extraction methods that capture particular characteristics—especially small arbitrary deviations—of a specular surface must be computed rather than a 3D surface model.

We propose a novel feature extraction method that computes specular characteristics by using a general decomposition framework. The grey-level image is separated into a stack of binary images and for each binary image a shape analysis of the connected components is performed; local shape properties are combined within each binary image and across all binary images by first order statistics. Finally, we obtain a feature vector that describes shape statistics of different reflections in a grey-level image.

[1] We, here, use the term *specular surface* to describe surfaces such as metal or plastic that reflect a large amount of light.

The idea of using a stack of binary images instead of using only a single image has already been proposed, e.g. in [20] or [112]. For an image I with k grey levels a stack of binary images $\mathcal{B} = \{I_\tau\}$ with $\tau \in \{1, 2, \ldots, k\}$ is created; a single binary image I_τ is computed by:

$$I_\tau(x, y) = \begin{cases} 1 & : \quad I(x, y) \geq \tau \\ 0 & : \quad I(x, y) < \tau \end{cases} \quad . \tag{3.1}$$

This decomposition is lossless, since the input image can always be recovered by summing up all binary images. Figure 3.1 demonstrates the decomposition of a grey-level image into a stack of binary images and the computation of image features. Each binary image I_τ is decomposed into a set of black and white components $\{\mathbf{C}^{\mathrm{w}}(\tau), \mathbf{C}^{\mathrm{b}}(\tau)\}$ with $\mathbf{C}^{\mathrm{w}}(\tau) = \{C_0^{\mathrm{w},\tau}, \ldots, C_{m-1}^{\mathrm{w},\tau}\}$ and $\mathbf{C}^{\mathrm{b}}(\tau) = \{C_0^{\mathrm{b},\tau}, \ldots, C_{n-1}^{\mathrm{b},\tau}\}$, where m and n are the numbers of black and white components. Each component $C_i^{*,\tau} = \{\boldsymbol{x}_k\}$ consists of pixel positions $\boldsymbol{x}_k \in \{0, 1, \ldots, H-1\} \times \{0, 1, \ldots, W-1\}$, where H and W are the height and the width of the input image. We omit, for convenience, the index and the colour of a component if these are unnecessary, and we use $C_i = C_i^{*,\tau}$ for abbreviation. The superscript "$*$" is used whenever both component colours are considered.

We employ a weight for each component C_i, which we define as its relative size compared to the total area of all components with the same colour:

$$\mathtt{PROP}\,(C_i) = \frac{\mathtt{AREA}\,(C_i)}{\sum_k \mathtt{AREA}\,(C_k)} \quad . \tag{3.2}$$

For capturing image characteristics that arise due to specular reflections, we use the decomposition scheme and evaluate local shape features. We, therefore, compute several general properties of each component such as eccentricity, compactness, or perimeter (see Table 3.1). We, then, obtain a feature vector $\boldsymbol{g}_i$ that describes shape as well as spatial properties for each black or white component C_i:

$$\boldsymbol{g}_i^* = (\mathtt{PERIM}, \mathtt{DISTC}, \ldots, \mathtt{REGAR})^{\mathrm{T}} \quad . \tag{3.3}$$

A detailed discussion on geometric shapes can be found in [87, Chapter 9], for instance.

Since these local features are computed for each component in the binary image, we must combine them to form a feature vector for all components in the binary image. We, hence, scale each feature g in two different ways; first, we compute the weighted mean using the component's relative size PROP (see Equation 3.2) and, second, we compute the standard mean, i.e. the sum scaled by the number of components (NOC). In case of the local feature PERIM, for instance, and the binary image I_τ, the two scalings are computed by:

$$\overline{\mathtt{PERIM}}(\tau) = \sum_k \mathtt{PERIM}(C_k)\ \mathtt{PROP}(C_k) \tag{3.4}$$

$$\overline{\mathtt{PERIM}}(\tau) = \frac{1}{\mathtt{NOC}(\tau)} \sum_k \mathtt{PERIM}(C_k) \quad . \tag{3.5}$$

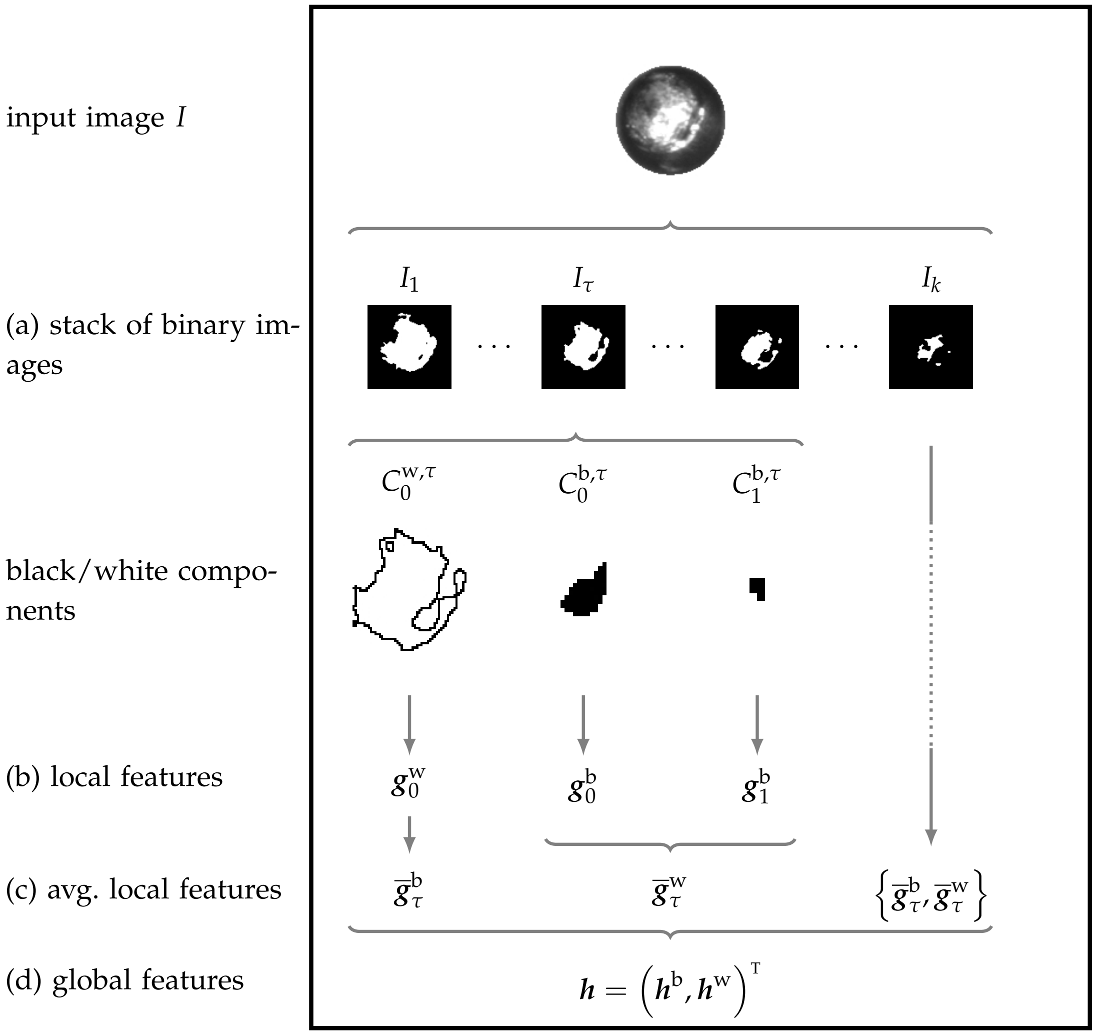


Figure 3.1.: The decomposition scheme for a grey-level input image I is composed of four stages; (a) the image is decomposed into a stack of binary images I_τ; (b) each binary image is separated into its black and white components, for which local features g, such as area or eccentricity, are computed; (c) local features are averaged according to the colour of the component; (d) global features h are computed through statistics over averaged local features.

Table 3.1.: Features for a component C; using a 4-neighbourhood two successive boundary points are denoted by $\boldsymbol{x}_m$ and $\boldsymbol{x}_{m+1}$; $\mu(C)$, μ_I are the centre of the component and the centre of the image, respectively; $F(\alpha)$ is the maximum distance between two boundary points when rotating the coordinate axis by $\alpha \in \mathcal{A} = \{0°, 5°, ..., 175°\}$; w_{BR} and h_{BR} are the width and height of the bounding rectangle; a, b are the major and minor axis of the ellipse that has the same second moment as the component.

feature $g(C)$	formula
perimeter (PERIM)	$\sum_m \|\boldsymbol{x}_m - \boldsymbol{x}_{m+1}\|_2$
distance from centre (DISTC)	$\|\mu(C) - \mu_I\|_2$
maximum Feret diameter (MAXFD)	$\max_{\alpha \in \mathcal{A}} F(\alpha)$
minimum Feret diameter (MINFD)	$\min_{\alpha \in \mathcal{A}} F(\alpha)$
mean Feret diameter (MEANF)	$\frac{1}{\|\mathcal{A}\|} \sum_{\alpha \in \mathcal{A}} F(\alpha)$
variance Feret diameter (VARFD)	$\frac{1}{\|\mathcal{A}\|} \sum_{\alpha \in \mathcal{A}} \left[F(\alpha) - \texttt{MEANF}\right]^2$
area of bounding rectangle (AREAB)	$w_{\text{BR}}\, h_{\text{BR}}$
eccentricity (ECCEN)	$\frac{\sqrt{a^2+b^2}}{a}$
aspect ratio (ASPAR)	$\frac{\texttt{MAXFD}(\tau)}{\texttt{MINFD}(\tau)}$
extent (EXTEN)	$\frac{\texttt{AREA}}{\texttt{AREAB}}$
formfactor (FORMF)	$\frac{4\,\pi\,\texttt{AREA}}{\texttt{PERIM}^2}$
roundness (ROUND)	$\frac{4\,\texttt{AREA}}{\pi\,\texttt{MAXFD}^2}$
compactness (COMPT)	$\frac{2\sqrt{\texttt{AREA}}}{\sqrt{\pi}\ \texttt{MAXFD}}$
regularity of aspect ratio (REGAR)	$\left[1 + \texttt{VARFD} + \texttt{MAXFD} - \texttt{MINFD}\right]^{-1}$

name	formula
`maximum`	$\max_\tau g(\tau)$
`minimum`	$\min_\tau g(\tau)$
`mean`	$\frac{1}{\lvert g(\tau)\rvert} \sum\limits_{g(\tau)} g(\tau)$
`variance`	$\frac{1}{\lvert g(\tau)\rvert} \sum\limits_{g(\tau)} (g(\tau) - \texttt{mean})^2$
`median`	$\arg\min_m E(\lvert g(\tau) - m\rvert)$
`sample mean`	$\frac{1}{\sum_\tau g(\tau)} \sum_\tau \tau\, g(\tau)$
`sample std.`	$\sqrt{\frac{\sum_\tau (\tau - \texttt{sample mean})^2 g(\tau)}{\sum_\tau g(\tau)}}$
`entropy`	$-\sum_{g(\tau)} p_{g(\tau)} \log p_{g(\tau)}$

Table 3.2: Global features, where $g(\tau)$ is one of the averaged local features described before and τ corresponds to the threshold for which the binary image has been computed.

With these two scalings, the area of each component as well as the total number of components is directly incorporated into each local feature.

So far, we obtain two feature vectors that capture averaged local characteristics of the binary image I_τ

$$\overline{\boldsymbol{g}}^*_\tau = (\overline{\texttt{PERIM}}(\tau), \overline{\texttt{DISTC}}(\tau), \ldots, \overline{\texttt{REGAR}}(\tau))^\mathrm{T} \ . \tag{3.6}$$

Now, we further merge local features by computing first order statistics described in Table 3.2; most of the statistics combine local shape features into characteristics that represent shape statistics across all grey-levels, independent of the particular threshold τ for which the binary image has been computed. In contrast, `sample mean` and `sample std.` take into account the threshold τ to compute the weighted mean and weighted standard deviation. More precisely, the feature vector that captures shape statistics of black/white components is defined as

$$\boldsymbol{h}^* = (\max_\tau \overline{\texttt{PERIM}}(\tau), \max_\tau \overline{\texttt{DISTC}}(\tau), \ldots, \max_\tau \overline{\texttt{REGAR}}(\tau), \min_\tau \overline{\texttt{PERIM}}(\tau), \ldots)^\mathrm{T} \ . \tag{3.7}$$

Finally, for a gey-level input image we obtain a feature vector $\boldsymbol{h} = (\boldsymbol{h}^\mathrm{b}, \boldsymbol{h}^\mathrm{w})^\mathrm{T}$ that contains 448 features composed of 28 local features for a single component (14 for a black component and 14 for a white component), 2 scaling methods (by the proportional size and by the total number of components) and 8 global statistics. With this feature vector we can, for instance, describe the variance of the extent of black and white components or the entropy of the formfactor of black and white components, and we can verify whether this corresponds to the physical shape of the object—see Part II.

3.3. Statistical Fourier Descriptors

In the previous section, we have proposed an approach for computing statistics of shapes, which can be used for capturing properties of specular reflections, for instance. However, the local features such as eccentricity, compactness, or perimeter have been developed specifically for the application of surface inspection; hence, this feature set may yield worse results when used in other applications. In this section, we propose a novel feature extraction approach that employs the same decomposition scheme as before; instead of evaluating specifically designed local shape features, we employ the Fourier descriptors [39] of the component's boundary.

Although Fourier descriptors have been used for over 30 years, they are still found to be a valid shape description tool. In several comparisons, the Fourier descriptor approach has proved to outperform most other boundary-based methods regarding accuracy and efficiency [52, 74, 118]. However, Fourier descriptors have not been applied to images with overlaying objects or to images with multiple grey-levels.

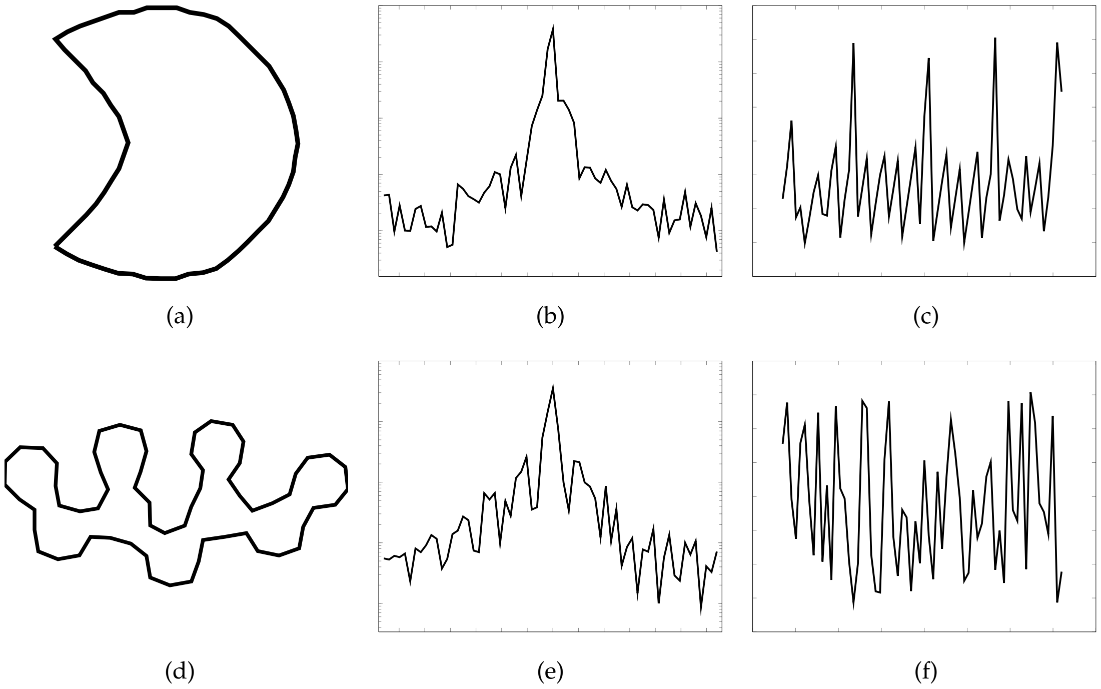
3.3.1. Fourier Descriptors

The boundary line of a two-dimensional object can be represented using a one-dimensional function $f(k)$, i.e. the shape signature, which can be obtained efficiently by combining the coordinates (x_k, y_k) of the boundary points $k = 0, ..., N-1$ to a complex number, i.e. $f(k) = x_k + j\,y_k$. The shape signature, however, must be periodic to be used for the 1-D discrete Fourier transform. In general, there are three different methods for obtaining a periodic shape signature: equal points sampling, equal angle sampling, and equal arc-length sampling; among these methods, the equal arc-length sampling has proven to achieve the best equal space effect [109]. Obviously, the number of sampling points N determines the accuracy of the approximation; a small number of sampling points, though, offer two advantages at the same time: the shape is smoothed and the Fourier transform is computed efficiently.

Since the shape signature is represented by a one-dimensional periodic signal, it can be transformed to the frequency domain using the discrete Fourier transform (DFT); the DFT of a shape signature $f(k)$ with N samples is given by

$$F_n = \sum_{k=0}^{N-1} f(k)\, e^{-j2\pi nk/N} \ , \quad 0 \leq n < N \ , \tag{3.8}$$

where F_n are the transform coefficients of $f(k)$ and known as Fourier descriptors. The Fourier descriptors are often expressed in polar form $F_n^* = |F_n|\, e^{j\phi_n}$ and several geometric transformations of the shape can be related to simple operations when transforming to the frequency domain. Translation of the shape, for instance, only affects the first Fourier coefficient or scaling the shape with a factor of a leads to a



(a) (b) (c)

(d) (e) (f)

Figure 3.2.: Two different shapes (1st column), magnitude $|F_n|$ (2nd column) and phase ϕ_n (3rd column) of the first 64 Fourier coefficients. Spectra are shifted such that the zero-component is located in the centre; axis are identical for each column. We can clearly recognise that the shapes differ in both magnitude as well as phase.

scaling of the Fourier coefficients by a. Furthermore, the coefficients can be normalised to be invariant towards the starting point by subtracting the phase of the second Fourier descriptor, weighted by n, from the phase of all Fourier descriptors:

$$F_n^* \to F_n^* \, e^{-j\phi_1 n} \quad . \tag{3.9}$$

Then, the starting point is approximately at angle 0. A detailed description and analysis of Fourier descriptors can be found in [50], for instance.

A common approach to shape analysis is to use only a subset of low-frequency coefficients; this captures relevant shape information and removes high frequency noise. In Figure 3.2 example objects are compared using the magnitude and phase of the first 64 Fourier descriptors. In most applications, we do not know about the shapes that might occur in the image and we, therefore, use both the magnitude as well as the phase of the Fourier descriptors.

3.3.2. Statistical Fourier Descriptors

Since the Fourier descriptor method can only be used for the contour of a single binary object, we propose a new Fourier-based method, called *statistical Fourier descriptor* (SFD), that describes shape statistics of multiple objects of different grey-levels. We, therefore, use the decomposition scheme presented in the previous section (see Figure 3.1). Instead of computing specifically designed local shape features, we compute the Fourier descriptors of the boundary of black and white components extracted from a single binary image. We simply replace the feature vector (Equation 3.3) that describes the characteristics of the *i*th black or white component by

$$\boldsymbol{g}_i^* = (|F_0|, \ldots, |F_{M-1}|, \phi_0, \ldots, \phi_{M-1})^{\mathrm{T}} \quad , \tag{3.10}$$

where $|F_i|$ are the magnitudes and ϕ_i the phases of the corresponding M Fourier descriptors, with $M \leq N$. We further combine the local shape features of N_c components in the binary image I_τ such that:

$$\overline{\boldsymbol{g}}_\tau^* = (\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{m}, \theta, \rho)^T \in \mathbb{R}^{3 \cdot 2M+2} \quad , \quad \text{with} \tag{3.11}$$

$$\mu_l = \frac{1}{N_c} \sum_{i=0}^{N_c-1} g_{il}^* \quad , \tag{3.12}$$

$$\sigma_l = \sqrt{\frac{1}{N_c} \sum_{i=0}^{N_c-1} \left(g_{il}^* - \mu_l\right)^2} \quad , \tag{3.13}$$

$$m_l = \max_i g_{il}^* \quad , \tag{3.14}$$

$$\boldsymbol{d} = \frac{1}{N_c} \sum_{i=0}^{N_c-1} (\boldsymbol{c}_c - \boldsymbol{c}_I) \quad , \tag{3.15}$$

where θ and ρ are the orientation and magnitude of the displacement vector $\boldsymbol{d}$, $\boldsymbol{c}_i$ is the centre of component i, and $\boldsymbol{c}_I$ is the image centre. With the properties of the displacement vector we can distinguish between binary images where the components are located circularly around the centre and binary images where the components are located at one particular side.

In the last step, we combine the local features of k binary images by calculating first order statistics such as mean, standard deviation, maximum, and sample mean to obtain a single feature vector $\boldsymbol{h} = (\boldsymbol{h}^{\mathrm{b}}, \boldsymbol{h}^{\mathrm{w}})^{\mathrm{T}} \in \mathbb{R}^{48M+16}$ for a given input image:

$$\boldsymbol{h}^{*} = (\boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{\varepsilon}, \boldsymbol{\eta})^{T} \in \mathbb{R}^{4\cdot(6M+2)} \quad , \tag{3.16}$$

$$\gamma_l = \frac{1}{k}\sum_{\tau=0}^{k-1} \overline{g}^{*}_{\tau\, l} \quad , \tag{3.17}$$

$$\delta_l = \sqrt{\frac{1}{k}\sum_{\tau=0}^{k-1} \left(\overline{g}^{*}_{\tau\, l} - \gamma_l\right)^{2}} \quad , \tag{3.18}$$

$$\varepsilon_l = \max_{\tau} \overline{g}^{*}_{\tau\, l} \quad , \tag{3.19}$$

$$\eta_l = \left(\sum_{\tau=0}^{k-1} \overline{g}^{*}_{\tau\, l}\right)^{-1} \sum_{\tau=0}^{k-1} \tau\, \overline{g}^{*}_{\tau\, l} \quad . \tag{3.20}$$

Then, for a given input image, the SFD approach computes a vector with $48M + 16$ features, where M is the number of used Fourier descriptors. For large M this feature vector becomes very high-dimensional and, thus, the performance of certain classifiers may degenerate; therefore, feature-selection methods can be applied to reduce the amount of features. Moreover, we also have to set the number of sampled boundary points N that are used for computing the Fourier transform. Since we are mostly interested in the low-frequent properties of the boundary's shape, we use small N, e.g. $N = 128$.

In Part II, we demonstrate that both specularity features as well as statistical Fourier descriptors yield accurate results in real applications such as the inspection of welding seams.

3.4. Radial Statistics

So far, we have presented feature extraction approaches that compute shape statistics based on a decomposition of the grey-level image into several binary images. In this section, we directly use raw intensities as input features. Since the number of image pixels is large, in most applications, we specifically arrange the intensities into segments and compute first order statistics for each segment to reduce the number of features significantly. This particular encoding of raw pixels is not only efficient, but can also be motivated biologically.

Neurophysiology studies have shown that visual stimuli are roughly represented in the retina as well as in the primary visual cortex by neighbouring regions, e.g. [23]. Based on these results so-called retinotopic maps have been computed, which illustrate

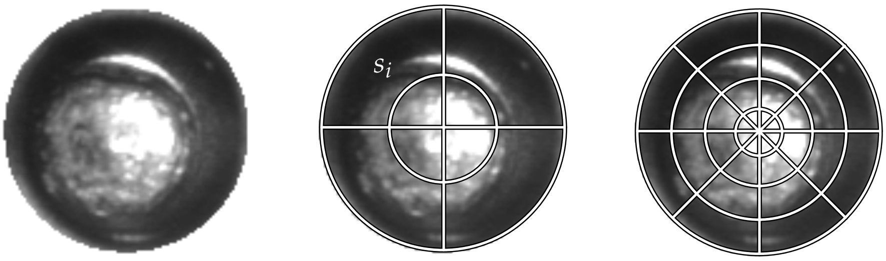


Figure 3.3.: Radial encoding exemplarily shown for a welding seam (left). A coarse radial grid will separate the image into few segments s_i (middle), whereas a fine radial grid yields many segments (right).

the spatial organisation of neurons that will respond to the same visual stimuli. Motivated by these observations, many methods in computer vision employ such retinotopic maps implicitly by using a retina-like radial grid (see Figure 3.3), e.g. [92, 35]; this radial grid is placed directly on the input image to radially encode raw intensities or it is applied to the results of image filtering, e.g., by a Gabor filter. Based on the radial separation, the mean value of each segment is computed, in most applications, and hence the feature vector contains the mean values of all segments.

Here, we present a more general framework for computing statistics of radially encoded raw pixel intensities to obtain a very low-dimensional feature vector. We radially divide the image into n segments s_i, compare Figure 3.3, for which we compute the mean of intensities

$$\mu_i = \frac{1}{|s_i|} \sum_{(x,y)\in s_i} I(x,y) \ . \tag{3.21}$$

We further compute first order statistics of these mean values such as maximum, mean, and standard deviation to analyse the mean values across all segments:

$$\begin{aligned} \beta_1 &= \max_i \ \mu_i \ , & (3.22) \\ \gamma_1 &= \min_i \ \mu_i \ , & (3.23) \\ \delta_1 &= \frac{1}{n}\sum_{i=1}^{n} \mu_i \ , & (3.24) \\ \varepsilon_1 &= \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\mu_i - \delta_1)^2} \ . & (3.25) \end{aligned}$$

These measures are combined into the feature vector

$$\boldsymbol{f}_1 = (\beta_1, \gamma_1, \delta_1, \varepsilon_1, \eta_1)^{\mathrm{T}} \ , \ \text{where} \tag{3.26}$$

$$\eta_1 = \frac{1}{n}\mathrm{card}(\mathcal{Z}_1) \ , \quad \text{and} \quad \mathcal{Z}_1 = \{\mu_i \,|\, \varepsilon_1 < |\mu_i - \delta_1|\} \ . \tag{3.27}$$

By η_1 we compute the fraction of segments for which the mean intensity is significantly larger compared to the mean value over all segment means. A large η_1 will indicate an irregular image texture, whereas a small η_1 will indicate a rather regular or homogeneous texture.

However, important image characteristics can be located at the border of two neighbouring segments such that the mean values vary only slightly. We, therefore, compute the standard deviation σ_i of each segment and evaluate the same measures as before:

$$\beta_2 = \max_i \sigma_i \ , \tag{3.28}$$

$$\gamma_2 = \min_i \sigma_i \ , \tag{3.29}$$

$$\delta_2 = \frac{1}{n}\sum_{i=1}^{n} \sigma_i \ , \tag{3.30}$$

$$\varepsilon_2 = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\sigma_i - \delta_2)^2} \ . \tag{3.31}$$

Again, we combine these measures to obtain the feature vector

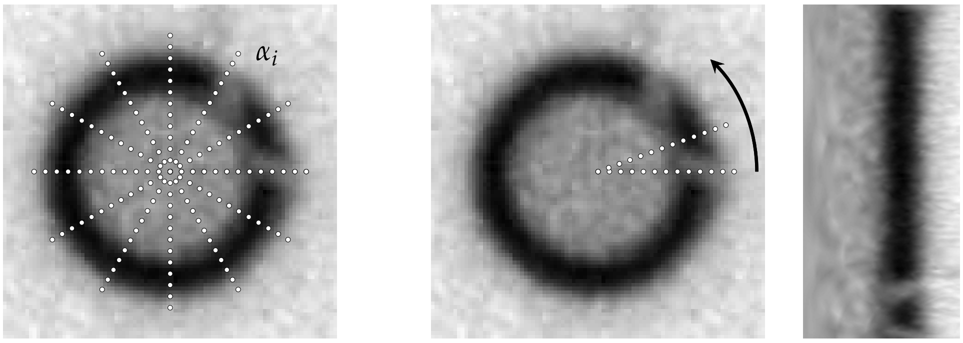
$$\boldsymbol{f}_2 = (\beta_2, \gamma_2, \delta_2, \varepsilon_2, \eta_2)^{\mathrm{T}} \ , \ \text{where} \tag{3.32}$$

$$\eta_2 = \frac{1}{n}\mathrm{card}(\mathcal{Z}_2) \ , \text{and} \quad \mathcal{Z}_2 = \{\sigma_i \,|\, \varepsilon_2 < |\sigma_i - \delta_2|\} \ . \tag{3.33}$$

Finally, we obtain a ten-dimensional feature vector $\boldsymbol{f} = (\boldsymbol{f}_1, \boldsymbol{f}_2)^{\mathrm{T}}$ for a given input image I. The only parameter we must choose is the number of segments the image is divided into.

Alternatively, the image can also be analysed by radial sampling, where the image is sampled for different orientations α_i and the intensities are computed by bilinear interpolation (see Figure 3.4). Compared to the previous technique, where we compute the mean of segments using every image pixel, here, the number of orientations α_i controls the number of involved pixels and, therefore, determines computational complexity. An advantage of this sampling technique is that it implicitly performs an image transform—the image is *unrolled* or transformed to polar coordinates (see Figure 3.4). Hence, we can use the transformed image to compute the features described in 3.26 and 3.32 column-wise. Figure 3.5 compares the number of involved pixels between the two radially encoded feature sets; for few orientations the sampling technique employs only a fraction of pixels compared to the separation into segments and it grows linearly in the number of orientations; at around 130–140 orientations both approaches use the same number of pixels; for a large number of orientations the sampling technique performs an oversampling and employs more pixels than image pixels.

We use these novel feature sets for the inspection of welding seams and compare with specularity features as well as statistical Fourier descriptors—see Part II.



(a) radial sampling

(b) radial "unrolling"

Figure 3.4.: Left: Image of a LED sampled radially with 32 directions α_i, i.e. angular resolution of 30 degrees; the intensities along each direction are computed by bilinear interpolation. Middle: Image is sampled radially (counter clockwise) with an angular resolution of 1 degree and a spatial resolution of 1 px (left). Right: Image after transformation; defects that affect the dark ring become more visible after transformation and features can be computed column-wise.

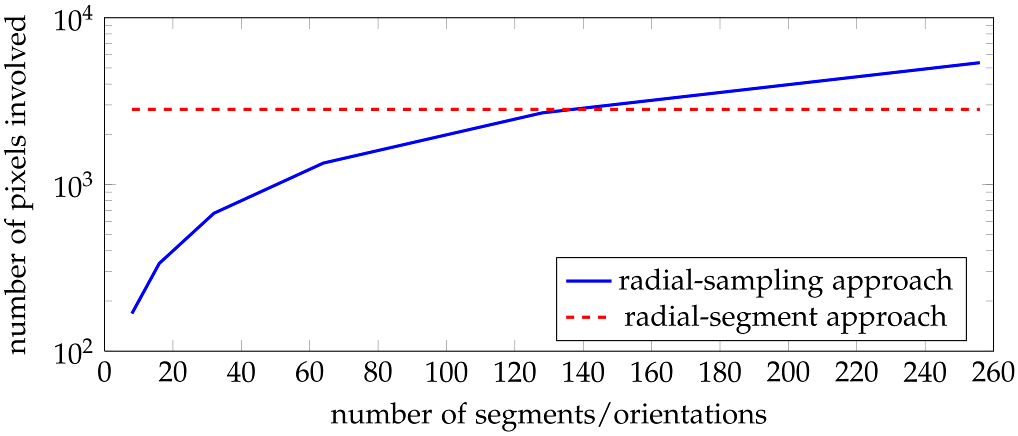


Figure 3.5.: Comparison of both feature-extraction methods concerning the number of pixels involved for different parameters, i.e. number of segments and number of orientations; we have used the welding seam image of Figure 3.3.

3.5. Local Weibull Image Statistics

So far, we have introduced feature extraction methods that either compute geometrical statistics or specifically encoded pixel intensities. In this section, we focus on the analysis of image gradient distributions in local image regions, and we demonstrate how characteristics of image gradients are described efficiently; these characteristics can be used, for instance, to detect small and subtle deviations in surface properties.

Recent studies have shown that natural image statistics as well as stochastic texture perception can be analysed by using a Weibull distribution. Geusebroek and Smeulders [36, 37], for example, found that the distribution of gradient magnitudes of 54 materials out of 61 in the Curet collection [27] consistently follow a Weibull distribution. Overall, they report that spatial image statistics change from a power-law to a normal distribution through the Weibull type distribution as the complexity of the scene increases. Therefore, Weibull distributions can be applied to image and texture analysis. Furthermore, it has been shown that Weibull image statistics yield accurate results for unsupervised image segmentation, image classification, and even for the analysis of visual content [116, 9]. A significant study has been conducted by Scholte et al. [91], in which they demonstrate that the distribution of contrast values in natural images generally form a Weibull distribution. More suprisingly, they found that parameters of the Weibull distribution strongly correlate with EEG responses of subjects viewing these images.

Although it seems reasonable to analyse the contrast in image patches by evaluating Weibull statistics, it remains unclear if Weibull statistics yield accurate results for describing small, subtle, and miscellaneous deviations in texture images.

We present a simple and non-parametric approach to feature extraction of texture images by using Weibull parameters. Figure 3.6 depicts the steps involved in this feature extraction approach; first, we extract local image patches of a given texture image, for which the patch size is automatically determined; second, for each local image patch we compute image gradients and the distribution of their magnitudes, for which we,

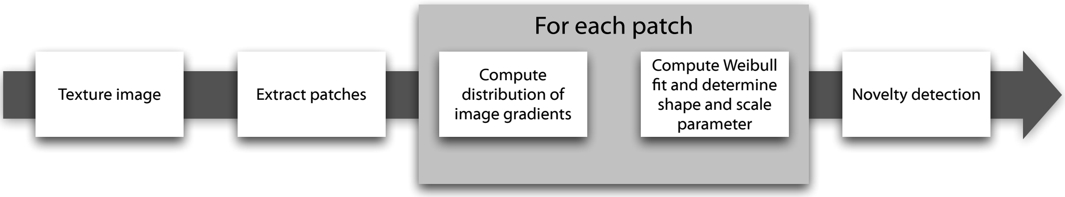

For each patch
Texture image
Extract patches
Compute distribution of image gradients
Compute Weibull fit and determine shape and scale parameter
Novelty detection

Figure 3.6.: Scheme for analysing texture images locally. Methods for novelty detection will be presented in the following chapter.

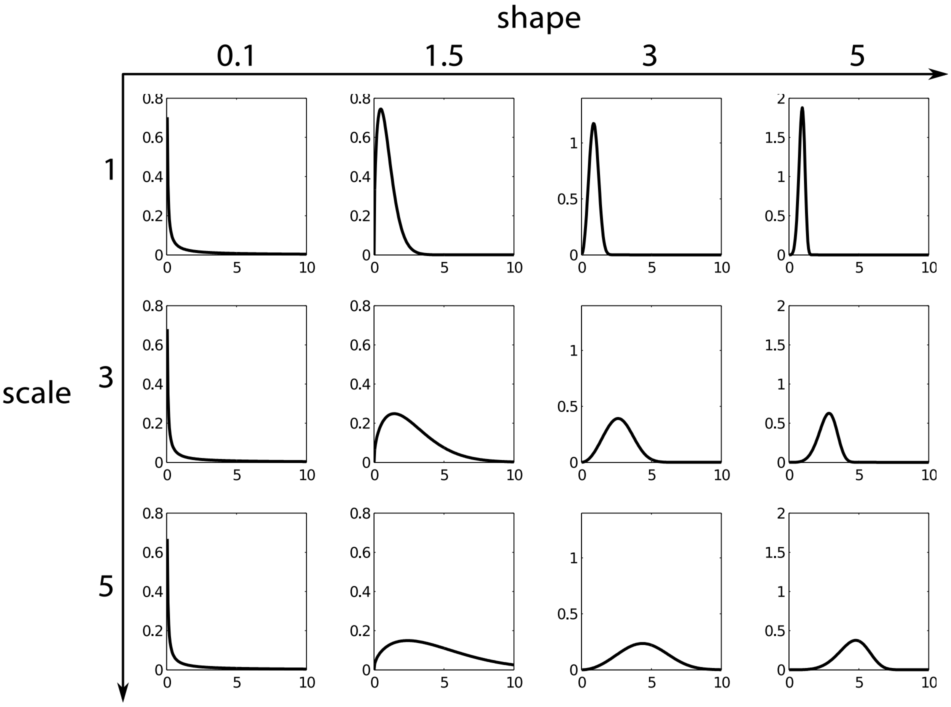


Figure 3.7.: Weibull distributions for different scales and shapes. A distribution similar to the power-law arises for small shape values (less than 1), whereas a Gaussian distribution is approximated by shape values around 3; intermediate shape values yield a typical Weibull type distribution.

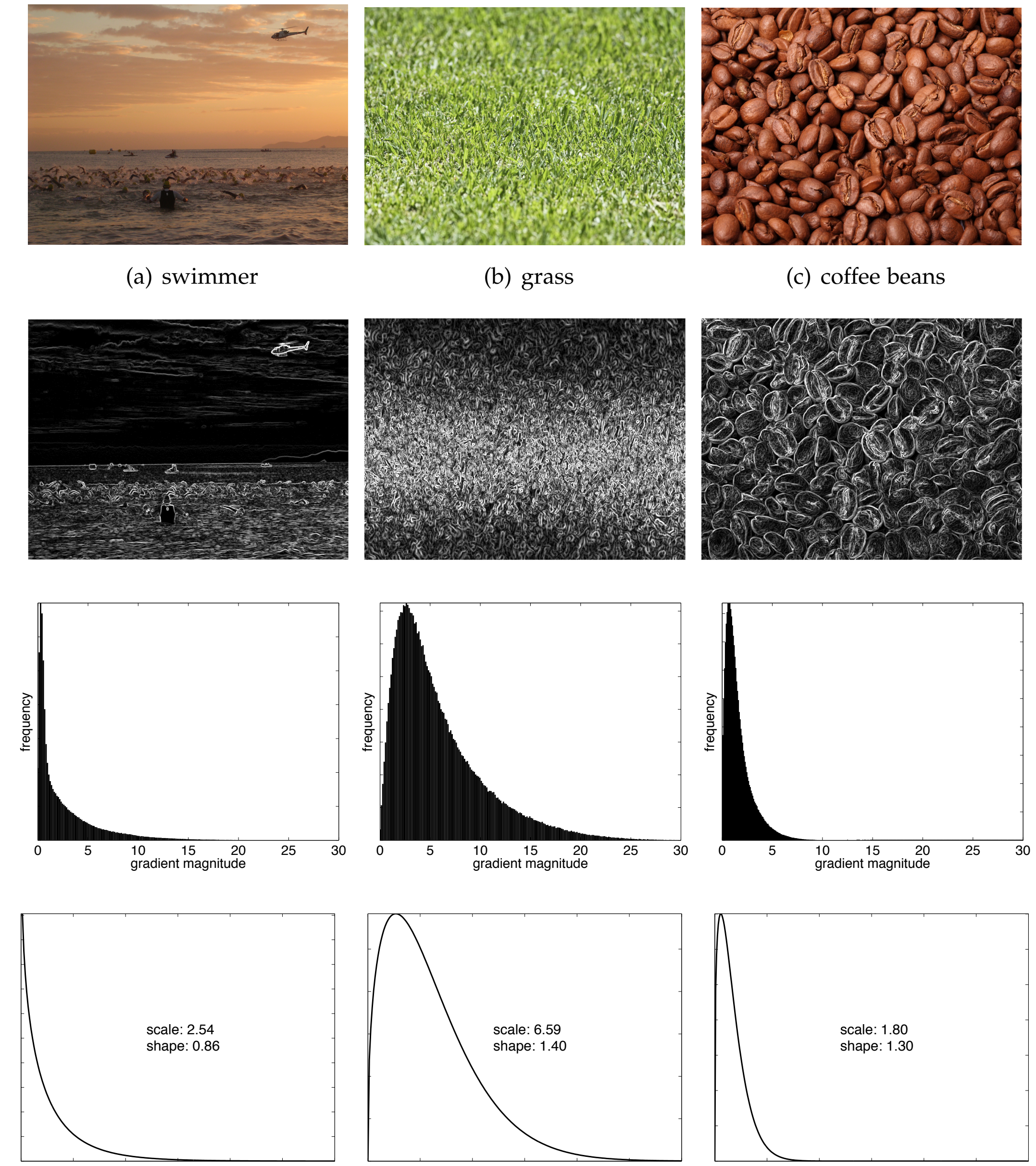
then, compute a Weibull fit and obtain the scale and shape parameter; finally, we apply a novelty-detection method using the two Weibull features. Note that novelty-detection methods will be introduced in the next chapter, here we focus on the feature extraction.

We will use the parametrized Weibull distribution with the following probability density function

$$p(x) = \frac{\beta}{\alpha}\left(\frac{x}{\alpha}\right)^{\beta-1} \mathrm{e}^{-\left(\frac{x}{\alpha}\right)^{\beta}} , \tag{3.34}$$

where $x > 0$ is the edge response of a Gaussian derivative filter, $\beta > 0$ the shape parameter, and $\alpha > 0$ the scale parameter. Figure 3.7 shows the Weibull distributions for different shape and scale parameters; apparently, the parametrized Weibull distribution captures a variety of different distributions such as power-law or Gaussian. Commonly, the shape and scale parameter are determined by maximum likelihood estimation [38] and used as image features, for instance, to analyse image content—see Figure 3.8.

Since our goal is to capture small deviations of texture images, we, in constrast, locally estimate the Weibull parameters of the distribution of edge responses, see Figure 3.9;


(a) swimmer

(b) grass

(c) coffee beans

Figure 3.8.: Example of three different natural images (first row) with significantly different visual appearance indicated by their Weibull parameters, shape and scale. The gradient magnitudes (second row) are used to compute a histogram (third row) from which the Weibull distribution is estimated (fourth row). Whereas the gradient magnitude histogram of the first image (swimmer) follows a power-law distribution, the magnitudes of the second (grass) and third image (coffee beans) follows a typical Weibull type distribution. Not only the different visual content of the first image and the other two images is described by significantly different Weibull parameters, but also the classification between fine textures (grass) and coarse textures (coffee beans) can be based on the Weibull parameters.

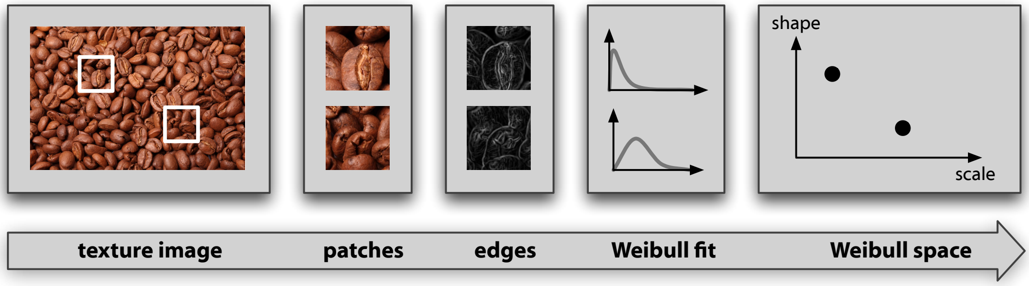


Figure 3.9.: Extraction of local texture characteristics. First, we extract local image patches and compute their gradient magnitudes. Second, the distribution of local gradient magnitudes is captured by fitting a Weibull distribution via maximum likelihood estimation. Finally, defective regions are detected by exploring the space that is determined by the Weibull parameters.

we apply first-order directional Gaussian derivative filters G_x, G_y to the image I

$$G_1 = \frac{\partial G(x,y)}{\partial x} \quad , \qquad G_2 = \frac{\partial G(x,y)}{\partial y} \quad , \qquad G(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{3.35}$$

to compute the gradient magnitude

$$|\nabla I(x,y)| = \sqrt{[I(x,y) \otimes G_1(x,y)]^2 + [I(x,y) \otimes G_2(x,y)]^2} \ . \tag{3.36}$$

By estimating the Weibull parameters of the distribution of local gradient magnitudes we obtain samples in the space of the shape and scale parameters. The basic idea is that within this space samples from defect-free image regions build clusters, whereas samples from defective regions significantly deviate from these clusters such that novelty-detection methods can be applied to detect these defective regions.

Figure 3.10 shows an example texture image with a subtle defect and samples in the Weibull space that are obtained by estimating the parameters of the Weibull fit from the gradient distribution of image patches. Apparently, most of the image patches form a large cluster, whereas few image patches are further from this cluster—these patches, obviously, belong to defective image regions. Although, in this example, texture features such as homogeneity or irregularity could also be computed to capture the defect, we will demonstrate in Part II of this thesis that local Weibull features can successfully be applied for various types of defects and various textures.

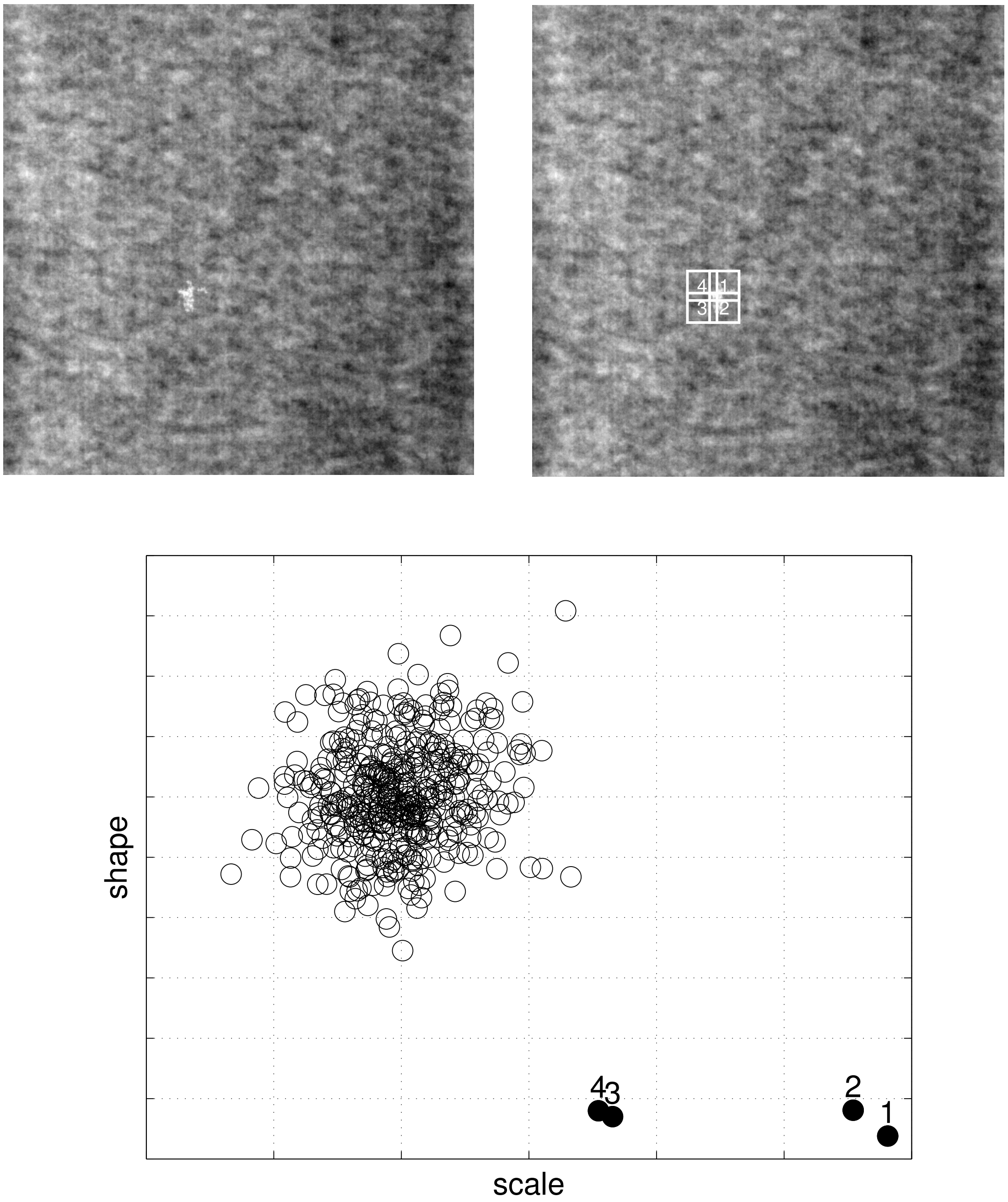
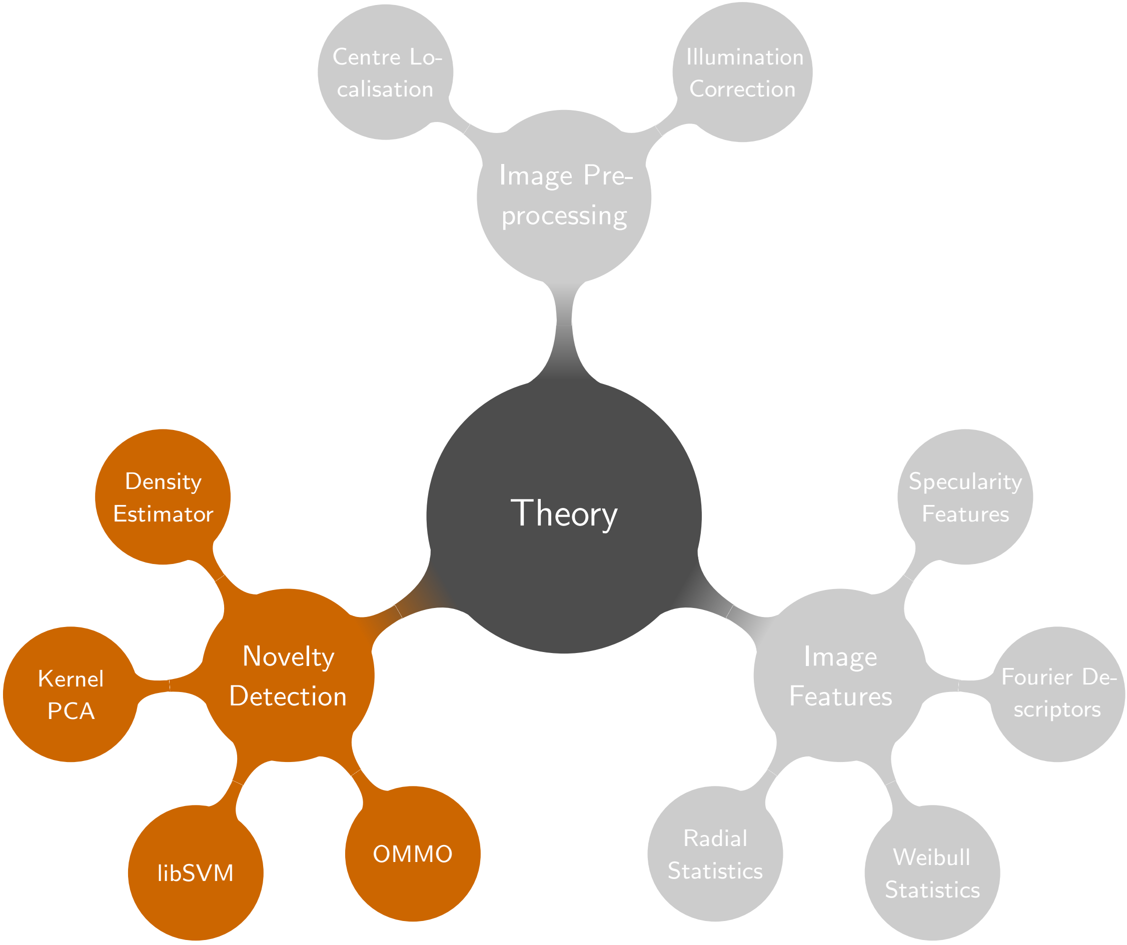


Figure 3.10.: Top left: Example texture image with a small, bright deviation near the image centre. Bottom: In the Weibull space a significant cluster can be detected. Bottom and top right: The four samples that are most far away from the cluster correspond to local image patches at the defective region.

4. Novelty Detection



Parts of this chapter are joint work with others. Kai Labusch and I came up almost simultaneously with the idea for a simple, incremental and support-vector based method for novelty detection. I refined that idea, implemented the algorithm, called OMMO, and derived the proofs. I came up with the idea of performing fast model selection for MaxMinOver-based learning algorithms and Sascha Klement and I contributed equally to refining and implementing that idea. Some of the work described in this chapter has been previously published in [64, 103]; minor parts of this chapter have been included in an article that is currently under review [98].

4.1. Introduction

Over recent years, the support vector machine [110, 24] has become a standard approach in solving pattern recognition tasks. Several training techniques, open-source toolboxes, and commercial libraries for computing support vector solutions exist, e.g. sequential minimisation optimisation (SMO) [81], LIBSVM [15], or MOSEK [5]. Although these methods are powerful and efficient, the details are diffcult to understand without a strong background in optimisation theory, and therefore they are difficult to motivate when explained to practitioners. Especially in industrial applications external libraries or toolboxes for solving optimisation problems are avoided, since source code is mostly unavailable and long-term external support is expensive. Hence, efficient, but simple learning algorithms that can easily be integrated and extended are required.

In many applications, one has to cope with the problem that only samples of one class are given; this class is often called *target* class. The task is to describe the target class and to separate it from the *outlier* class, which consists of all outliers; methods that solve this task are called novelty-detection methods. Either only few samples of the outlier class are given or outlier samples are missing completely. In these cases two-class classifiers often show poor generalisation performance and, hence, it is advantageous to employ novelty detection methods. Especially in biomedical applications such as cancer detection or tissue classification, where samples from healthy patients are frequent but negative samples are rare, novelty-detection methods have become very important. Moreover, novelty detection can even be applied for detecting outliers that have never been seen before, which is required in industrial applications, when only defect-free samples can be described well, but various new types of defects can occur.

Approaches to novelty detection can roughly be divided into three groups: density-estimation methods, reconstruction methods, and boundary methods. The first and the second group are the most powerful, because they derive a model of the data that is defined everywhere in the input space. In contrast, boundary methods consider an easier problem, that is, describing only the class boundary, instead of describing the complete data distribution. Markou and Singh [72, 73] published a comprehensive survey of novelty-detection methods, where they discuss statistical approaches in the first part and neural network approaches in the second part.

In this chapter, we describe a novel, simple, and incremental boundary method based on the support vector approach. First, we show theoretically that our method is comparable with state-of-the-art support-vector methods for novelty detection and we demonstrate that our method yields efficient solutions. Then, we demonstrate its performance and efficiency by comprehensive experiments on several benchmark datasets. Our novelty-detection method provides state-of-the-art performance despite being extremely simple and therefore useful especially for practitioners who are not within the field of machine learning.

4.2. Previous Work

4.2.1. Kernel Density Estimation

One of the most important non-parametric approaches for novelty detection is the so-called kernel density estimator (KDE) [30] or Parzen estimator [80] that estimates the underlying density. If the estimated density of a sample is below a particular threshold, this sample is considered as outlier; the threshold must be chosen to meet the required specification or can be computed automatically by analysing, for instance, the corresponding receiver-operator characteristics. Given N data samples $\boldsymbol{x}_i \in \mathbb{R}^D$, the estimated probability density function at $\boldsymbol{x}$ can be written as

$$p(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{m} k\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right) \quad , \tag{4.1}$$

where h is a smoothing parameter, $k : \mathbb{R}^D \to \mathbb{R}$ some kernel function, and m a normalisation factor such that

$$\int p(\boldsymbol{x}) \, d\boldsymbol{x} = 1 \quad . \tag{4.2}$$

A common choice for the kernel function is the Gaussian, which yields the following kernel density model

$$p(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{(2\,\pi\,\sigma^2)^{D/2}} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2\,\sigma^2}\right) \quad , \tag{4.3}$$

where σ represents the standard deviation of the Gaussian and D the dimension of the input space. The density model is obtained by placing a Gaussian over each data point, summing up the contributions over all data samples, and dividing by the number of samples for correct normalisation. Figure 4.1 shows the application of the density model to toy data with different values of σ; apparently, this parameter strongly influences the estimated model—it yields smooth density models for large σ and very sensitive models for small σ. There are several techniques for computing appropriate σ such as maximisation of the log likelihood or cross validation if some defect samples are available for training.

On the one hand, KDE does not require any training phase, which is beneficial in real-world applications when new samples become available and the model must be updated, but on the other hand KDE requires storage of the whole dataset; moreover, the cost of evaluating the density of a new sample grows linearly with the size of the dataset. There are many techniques that achieve a good approximation of the density with only a limited number of data samples such as reduced set methods or clustering methods. These optimisations, however, introduce new parameters, which must be validated and which strongly influence overall performance.

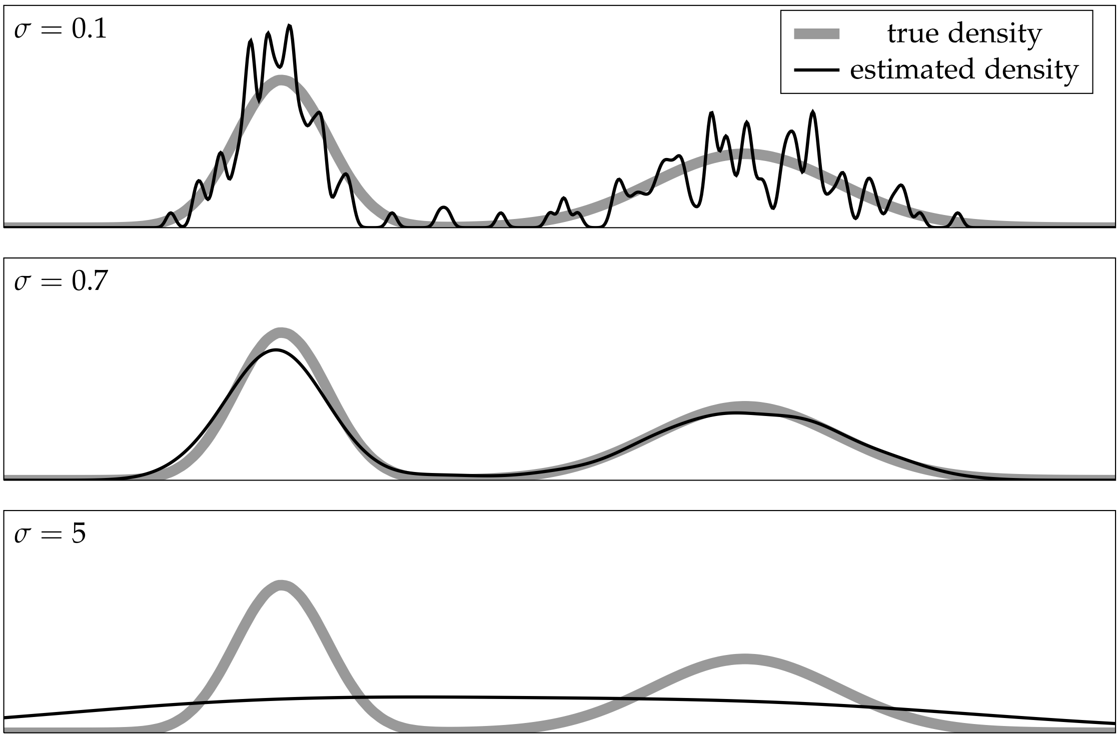


Figure 4.1.: Estimated density model for toy data and varied bandwidth. Top: For small σ the estimated model is very sensitive. Bottom: A large σ results in a smoothed model such that the bimodal nature of the underlying distribution is washed out. Middle: A good density model is obtained for some intermediate σ.

4.2.2. Kernel Principal Component Analysis

Principal component analysis (PCA) has become a standard tool for various applications such as data analysis, feature extraction, or feature selection. If the input variables are linearly related to each other, the direction with maximum variance describes the data best and is therefore called first principal component (see Figure 4.2). If the input variables are nonlinearly related to each other, the data samples are transformed from the input space into a feature space of higher dimensionality. It is assumed that within this feature space the data samples are then linearly related and standard principal component analysis can be performed—in this case, it is called *kernel* principal component analysis (KPCA) [89]. Since in most cases the data distribution is unknown, it is unclear whether the input variables are linearly related to each other.

Hoffmann [46] recently described how KPCA can also be used for novelty detection; there, a sample is considered as novel if its reconstruction error for a particular set of principal components exceeds a threshold. The density of the reconstruction error is modelled in the whole input space and therefore this approach can be interpreted as

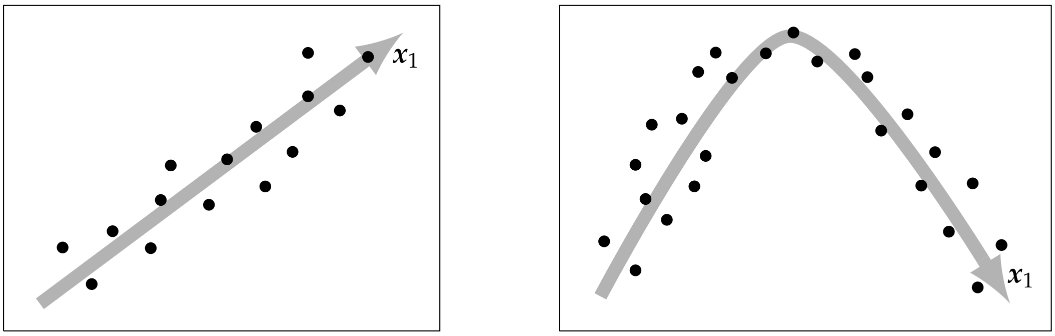


Figure 4.2.: The first principal component x_1 for exemplary data samples in the linear case (left) and in the nonlinear case (right).

a special case of a density method for novelty detection—even though the common properties such as normalisation, see Equation 4.2, for instance, are not necessarily fulfilled.

In the following we will describe the KPCA method and how it can be used as novelty detector. We denote the transformation function from the input space $\mathcal{X}$ to a feature space $\mathcal{H}$ by

$$\phi : \mathcal{X} \rightarrow \mathcal{H} \quad , \quad \boldsymbol{x} \rightarrow \phi(\boldsymbol{x}) \ . \tag{4.4}$$

Then, the covariance matrix of the samples $\boldsymbol{x}_i$, $(i = 1, \ldots, N)$, in feature space can be written as

$$\boldsymbol{C} = \frac{1}{N} \sum_{i=1}^{N} \hat{\phi}(\boldsymbol{x}_i) \hat{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} \ , \tag{4.5}$$

where

$$\hat{\phi}(\boldsymbol{x}_i) = \phi(\boldsymbol{x}_i) - \phi_0 = \phi(\boldsymbol{x}_i) - \frac{1}{N} \sum_{i=1}^{N} \phi(\boldsymbol{x}_i) \ . \tag{4.6}$$

We assume, for a moment, that the Eigenvectors can be written as a linear combination of the data samples in features space such that

$$\boldsymbol{v} = \sum_{i=1}^{N} \alpha_i \phi(\boldsymbol{x}_i) \ . \tag{4.7}$$

We, then, obtain the Eigenvectors and Eigenvalues by solving the Eigenvalue problem

$$n \lambda \alpha = \boldsymbol{K} \alpha \ , \tag{4.8}$$

where $\boldsymbol{K}_{ij} := \hat{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} \hat{\phi}(\boldsymbol{x}_j)$ is a kernel matrix centred in the feature space. A detailed derivation of the Eigenvalue problem and the centred kernel matrix are given in Appendix A. After computing the Eigenvectors, we can project a data sample $\phi(\boldsymbol{z})$ onto the l-th Eigenvector $\boldsymbol{v}_l = \sum_{i=1}^{N} \alpha_i^l \hat{\phi}(\boldsymbol{x}_i)$ in feature space by

$$\begin{aligned}
p_l(\boldsymbol{z}) &= (\phi(\boldsymbol{z}) - \phi_0)^{\mathrm{T}} \boldsymbol{v}_l \\
&= (\phi(\boldsymbol{z}) - \phi_0)^{\mathrm{T}} \left(\sum_{i=1}^{N} \alpha_i^l (\phi(\boldsymbol{x}_i) - \phi_0) \right) \\
&= \sum_{i=1}^{N} \alpha_i^l \left(K(\boldsymbol{z}, \boldsymbol{x}_i) - \frac{1}{N} \sum_{j=1}^{N} K(\boldsymbol{z}, \boldsymbol{x}_j) - \frac{1}{N} \sum_{k=1}^{N} K(\boldsymbol{x}_k, \boldsymbol{x}_i) + \mu \right) \\
&= (\boldsymbol{\alpha}^l)^{\mathrm{T}} \left(\boldsymbol{K}' - \frac{\boldsymbol{K}\boldsymbol{e}}{N} \right) + (\mu - \mu_z)\, \boldsymbol{e}^{\mathrm{T}} \boldsymbol{\alpha}^l \quad ,
\end{aligned} \tag{4.9}$$

where $\boldsymbol{K}'_i = K(\boldsymbol{z}, \boldsymbol{x}_i)$, $\mu_z = \frac{1}{N} \sum_{j=1}^{N} K(\boldsymbol{z}, \boldsymbol{x}_j)$, $\mu = \frac{1}{N^2} \sum_{j,k=1}^{N} K(\boldsymbol{x}_j, \boldsymbol{x}_k)$, and $\boldsymbol{e} = (1, \ldots, 1)^{\mathrm{T}}$.

Let $\boldsymbol{V}$ be a matrix of row vectors that are the solution of Problem 4.8, i.e. q Eigenvectors $\boldsymbol{v}_j$; we define the projection of a sample $\phi(\boldsymbol{z})$ onto these q Eigenvectors as

$$\boldsymbol{p}_z := (p_1(\boldsymbol{z}), \ldots, p_q(\boldsymbol{z}))^{\mathrm{T}} = \boldsymbol{V} \phi(\boldsymbol{z}) \quad . \tag{4.10}$$

We can now use the reconstruction error $r(\boldsymbol{z})$ of a data sample $\boldsymbol{z}$ in feature space as novelty measure (see Figure 4.3), that is

$$\begin{aligned}
r(\boldsymbol{z})^2 &= \|\hat{\phi}(\boldsymbol{z}) - \boldsymbol{V}^{\mathrm{T}} \boldsymbol{V} \hat{\phi}(\boldsymbol{z})\|_2^2 \\
&= \|\hat{\phi}(\boldsymbol{z})\|_2^2 - 2\, \hat{\phi}(\boldsymbol{z})^{\mathrm{T}} \boldsymbol{V}^{\mathrm{T}} \boldsymbol{V} \hat{\phi}(\boldsymbol{z}) + \hat{\phi}(\boldsymbol{z})^{\mathrm{T}} \boldsymbol{V}^{\mathrm{T}} \boldsymbol{V} \boldsymbol{V}^{\mathrm{T}} \boldsymbol{V} \hat{\phi}(\boldsymbol{z}) \\
&= \|\hat{\phi}(\boldsymbol{z})\|_2^2 - \|\boldsymbol{V} \hat{\phi}(\boldsymbol{z})\|_2^2 \\
&= \|\phi(\boldsymbol{z}) - \phi_0\|_2^2 - \|\boldsymbol{V}(\phi(\boldsymbol{z}) - \phi_0)\|_2^2 \\
&= K(\boldsymbol{z}, \boldsymbol{z}) - \frac{2}{N} \sum_{i=1}^{N} K(\boldsymbol{z}, \boldsymbol{x}_i) + \frac{1}{N^2} \sum_{j,k=1}^{N} K(\boldsymbol{x}_j, \boldsymbol{x}_k) - \|\boldsymbol{p}_z\|_2^2 \\
&= K(\boldsymbol{z}, \boldsymbol{z}) - 2\, \mu_z + \mu - \boldsymbol{p}_z^{\mathrm{T}} \boldsymbol{p}_z \quad ,
\end{aligned} \tag{4.11}$$

where we used $\boldsymbol{V}\boldsymbol{V}^{\mathrm{T}} = \mathbf{1}$.

Only a fraction of Eigenvectors $\boldsymbol{v}_k$ are commonly used for further computations; we, therefore, normalise the Eigenvalues λ_k such that $\sum_{i=1}^{N} \lambda_i = 1$ and sort them in descending order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. Let $\tau \in (0, 1]$ be a user-defined threshold describing the amount of information that should be kept with respect to all Eigenvalues. We compute the cumulated sum of sorted Eigenvalues and preserve all Eigenvalues and corresponding Eigenvectors that are necessary to achieve threshold τ.

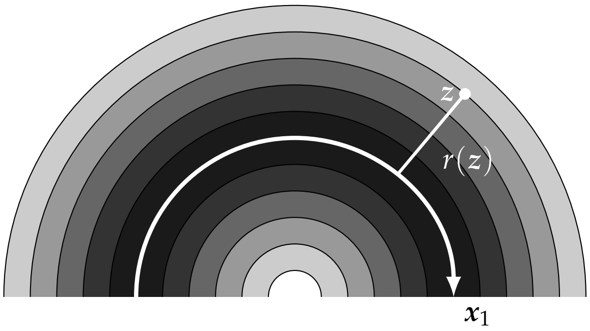


Figure 4.3.: The construction error $r(z)$ of a data point z using the first principal component x_1. Sample contour lines with equal reconstruction errors are shown as solid lines exemplarily; shades of grey roughly depict the different levels of reconstruction error.

Finally, the KPCA algorithm for novelty detection can be summarised as:

1 Compute centered and non-centered kernel matrices $\boldsymbol{K}$, $\hat{\boldsymbol{K}}$ according to Equation A.11.

2 Compute Eigenvalues and Eigenvectors $\boldsymbol{\alpha}^l$ by solving Problem 4.8.

3 Normalise Eigenvectors according to Equation A.8.

4 Choose a set of Eigenvectors.

5 Compute the novelty of a new sample $\boldsymbol{z}$ with Equation 4.11.

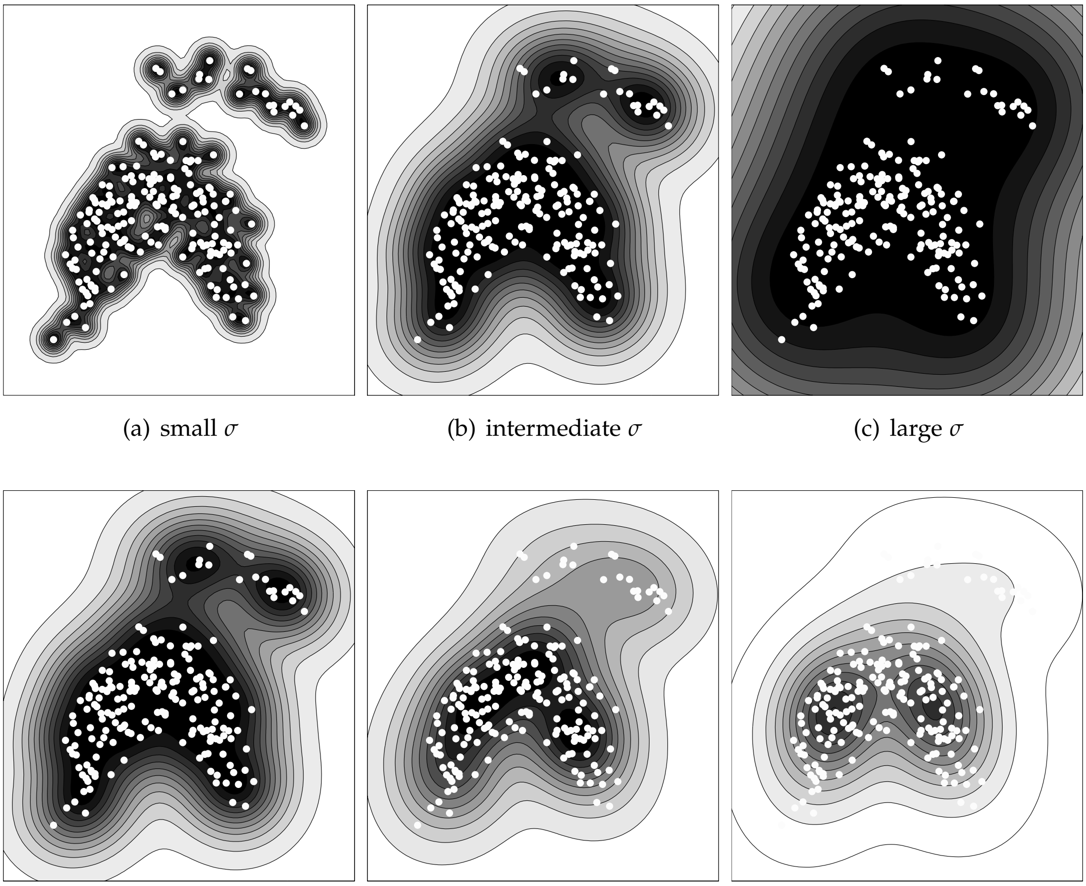
The kernel matrix $\boldsymbol{K}$ is computed by a user-defined kernel function k which is, in most cases, a Gaussian: $\boldsymbol{K}_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$. As for the kernel density estimator, the kernel parameter σ and the threshold τ are evaluated via cross validation. Figure 4.4 shows the results of applying the KPCA method to toy data and with different σ and τ.

4.2.3. Boundary Support Vector Methods

Several boundary methods for novelty detection have been developed, among which two have been introduced almost simultaneously—the approach by Tax and Duin [96, 97] and the approach by Schölkopf et al. [88].

Tax and Duin proposed a method, which is called *support vector data description* (SVDD), for finding the smallest enclosing hypersphere of given data samples $\boldsymbol{x}_i \in \mathcal{X}, (i = 1, \ldots, N)$; the hypersphere is described by radius R and centre $\boldsymbol{w}$, which can both be computed by solving the following optimisation problem:

$$\min_{\boldsymbol{w},R} \left(R + C \sum_i \xi_i \right) \quad \text{s.t.} \quad \forall i : \|\boldsymbol{x}_i - \boldsymbol{w}\| \leq R + \xi_i \ , \ \ \xi_i \geq 0 \ . \tag{4.12}$$



(a) small σ

(b) intermediate σ

(c) large σ

(d) large τ

(e) intermediate τ

(f) small τ

Figure 4.4.: The KPCA method applied to toy data and with a Gaussian kernel with parameter σ. Different shades of grey correspond to the reconstruction error; small errors are dark, whereas large errors are bright. Top row: Models with different σ and large τ; for small σ the solution is very sensitive (a), whereas a large σ results in a very smooth model that cannot accurately capture the details of the data (c); a good model is obtained for some intermediate σ (b). Bottom row: Models with different τ and intermediate σ; large τ yield a model that tries to cover as many data sample as possible (d); as τ decreases the model covers only regions with high data density (e); for small σ most of the data samples yield the same large reconstruction error and thus the solution describes only a small fraction of data samples accurately. Note that all axes have equal scaling.

The basic idea of this approach is to compute a hypersphere that describes the target class, that has minimal volume, and that is determined by only few data samples. Slack variables ξ_i are introduced to allow for samples that are located outside the hypersphere. By setting the partial derivatives of the corresponding Lagrangian to zero and resubstituting we obtain the dual optimisation problem

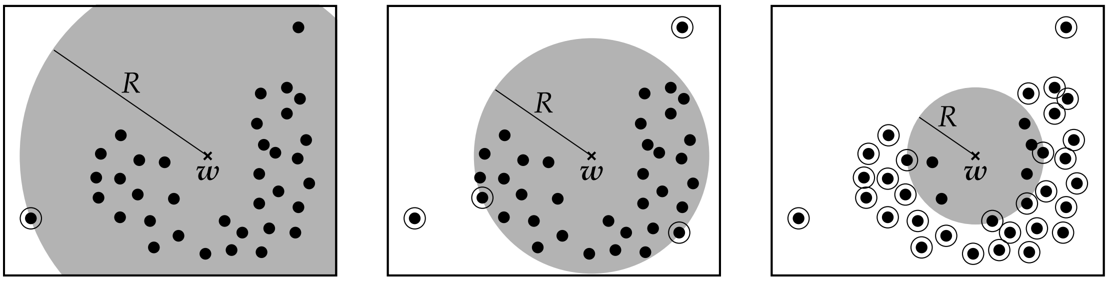
$$\min_{\boldsymbol{\alpha}} \left(\sum_{i,j} \alpha_i \alpha_j \, \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j - \sum_i \alpha_i \, \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_i \right) \quad \text{s.t.} \quad \forall i : 0 \leq \alpha_i \leq C \, , \; \sum_i \alpha_i = 1 \ . \tag{4.13}$$

The parameter $C \in [1/n, 1]$ controls the number of ”outliers“; for $C < 1/n$ no solution can be obtained, since the constraint $\sum_i \alpha_i = 1$ cannot be fulfilled, whereas for $C > 1$ a solution can always be found. A large fraction of samples will be outside the hypersphere for small C, whereas the number of data samples inside the hypersphere increases as C increases, see Figure 4.5.

Note that for the Lagrangian new constraints are obtained, e.g. the centre of the sphere must be a linear combination of the data samples

$$\boldsymbol{w} = \sum_i \alpha_i \boldsymbol{x}_i \ . \tag{4.14}$$

Since $\boldsymbol{w}$ is only determined by samples $\boldsymbol{x}_i$ with $\alpha_i > 0$, these samples are called *support vectors* (see Figure 4.5). According to the Karush-Kuhn-Tucker complementarity conditions, $0 < \alpha_i < C$ holds for support vectors on the boundary, whereas support vectors outside the hypersphere yield $\alpha_i = C$. The radius R of the sphere can thus be obtained by computing the distance between the centre and a support vector on the boundary. A new sample $\boldsymbol{x}$ is classified by comparing the distance to the centre of the



(a) large C (b) intermediate C (c) small C

○ support vectors
• data samples

Figure 4.5.: Example of the SVDD method applied to toy data. Left: For large C the hypersphere captures all data samples. Right: A small C results in a small hypersphere such that only a few samples lie inside. Middle: A good solution is obtained for some intermediate C.

sphere and R. In terms of support vectors this becomes

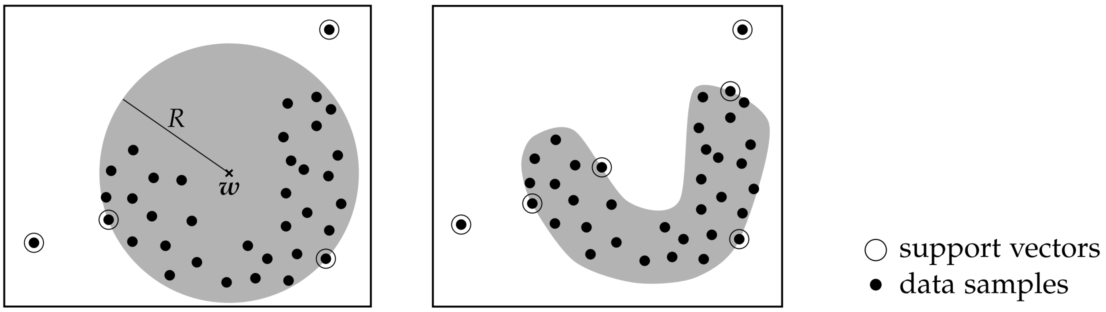
$$f(\boldsymbol{x}) = \operatorname{sgn}\left(R^2 - \|\boldsymbol{x} - \boldsymbol{w}\|^2\right) = \operatorname{sgn}\left(R^2 - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x} + 2\sum_i \alpha_i \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x} - \sum_{i,j} \alpha_i \alpha_j \, \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_i\right) \quad . \tag{4.15}$$

Since, in most cases, the data samples are not spherically distributed, the boundary must be more flexible to be applied to various data distributions. We can achieve such a flexible and powerful boundary by replacing the inner products $(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j)$ with a kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$, where ϕ maps the data samples into some high-dimensional feature space, in which Problem 4.13 can be solved. Since $\boldsymbol{x}_i$ is solely involved in inner products, we do not need to compute the mapping $\phi(\boldsymbol{x}_i)$ explicitly, instead, the mapping is used implicitly in the dot products—this is frequently referred to as the *kernel trick*. More details regarding kernels can be found in [25, chapter 3], for example. Note that in case of a kernel function, $\boldsymbol{w}$ can only be evaluated if the mapping ϕ can be computed explicitly; for some kernel functions ϕ only an approximation can be computed, for instance, by using pre-image techniques.

Although several kernel functions have been proposed, especially for two-class support vector classification, kernel functions that are based on only the distance of two data samples rather than on their inner product are suitable for SVDD. Hence, a Gaussian kernel

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|/2\sigma^2\right) \tag{4.16}$$

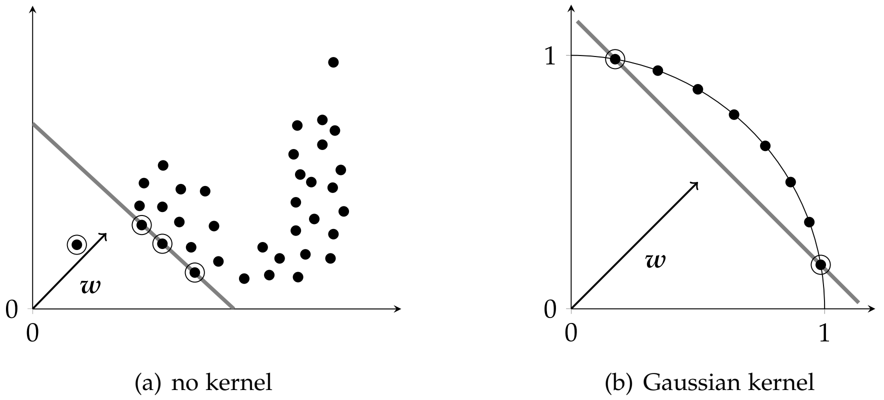
is applied in cases where the data do not follow a uniform circular distribution (see Figure 4.6). A Gaussian kernel, however, carries the risk of under- and overfitting if the kernel parameters are chosen inappropriately. To overcome this problem, cross-validation techniques are often applied to automatically determine appropriate kernel parameters.



(a) no kernel

(b) Gaussian kernel

Figure 4.6.: The SVDD method applied to toy data; if the data does not follow a uniform circular distribution a hypersphere yields an inaccurate data description (left), whereas by including a kernel, such as the Gaussian, one obtains a flexible and accurate data description (right).



(a) no kernel

(b) Gaussian kernel

Figure 4.7.: The method proposed by Schölkopf et al. computes a linear separation of the data samples from the origin. Left: Without a kernel the hyperplane linearly separates the data samples from the origin in the input space. Right: The Gaussian kernel maps the data samples onto the unit sphere in some higher-dimensional feature space, where the maximum-margin hyperplane is computed; this separation then corresponds to a non-linear class boundary in the input space, such as in Figure 4.6(b). Note that for the Gaussian kernel only a two-dimensional projection of the data samples and the separating hyperplane is shown.

Schölkopf et al. [88] proposed a method, which also belongs to the class of boundary methods; this method is closely connected to the previous method, as we will discuss later. They show that data description or estimating the support of a high-dimensional distribution, as it is called in their article, can be interpreted as two-class classification problem where the outlier class is represented by the origin (see Figure 4.7). They consider the problem of finding the hyperplane that separates the data samples from the origin with maximum distance, which can be formulated as the following optimisation problem:

$$\min_{w,\xi,\rho} \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{\nu l}\sum_i \xi_i - \rho \right) \quad \text{s.t.} \quad \forall i : \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i \geq \rho - \xi_i \ , \ \ \xi_i \geq 0 \ . \tag{4.17}$$

Here, $\nu \in (0, 1]$ is a regularisation parameter similar to C for the SVDD approach (see Equation 4.12). Again, the soft-margin problem is shown, which allows for misclassified samples by incorporating slack variables ξ_i. For small ν one obtains the hard-margin solution that enforces correct classification of most training samples, whereas for large ν many training samples are on the other side of the hyperplane.

Again, setting the partial derivatives of the corresponding Lagrangian equal to zero yields an expansion of $\boldsymbol{w}$ in terms of support vectors and upper bounds for the

Lagrangian multipliers α_i, β_i:

$$\boldsymbol{w} = \sum_i \alpha_i \boldsymbol{x}_i \tag{4.18}$$

$$\alpha_i = \frac{1}{\nu N} - \beta_i \leq \frac{1}{\nu N} \quad , \qquad \sum_i \alpha_i = 1 \ . \tag{4.19}$$

Hence, the dual problem is

$$\min_{\boldsymbol{\alpha}} \left(\sum_{i,j} \alpha_i \alpha_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \right) \quad \text{s.t.} \quad \forall i : 0 \leq \alpha_i \leq \frac{1}{\nu N} \ , \ \sum_i \alpha_i = 1 \ , \tag{4.20}$$

and the decision function becomes

$$f(\boldsymbol{x}) = \operatorname{sgn}\left(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} - \rho\right) = \operatorname{sgn}\left(\sum_i \alpha_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x} - \rho \right) \ , \tag{4.21}$$

where ρ is recovered by any support vector $\boldsymbol{x}_j$ on the boundary

$$\rho = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_j = \sum_i \alpha_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \ , \quad \forall \boldsymbol{x}_j : \ 0 < \alpha_j < \frac{1}{\nu N} \ . \tag{4.22}$$

Note that the major difference between the duals 4.13 and 4.20 is the linear term $\sum_i \alpha_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_i$. Moreover, 4.13 and 4.20 turn out to be equivalent, if all $\boldsymbol{x}_i$ lie on the surface of a sphere such that $\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_i$ is constant, which is satisfied for every kernel function that depends on only $\boldsymbol{x}_i - \boldsymbol{x}_j$. In the following, we assume that the data samples can be linearly separated from the origin, which is satisfied in case of a Gaussian kernel, for instance, since all data samples lie in the same orthant and have unit length; a detailed discussion can be found in [88, appendix].

4.3. OneClassMaxMinOver

In this section, we introduce a straightforward, incremental, and efficient algorithm, called *OneClassMaxMinOver* (OMMO), for support-vector data description; it is closely connected to the previous optimisation problems. We, therefore, rewrite the primal problem 4.17, proposed by Schölkopf et al., for the hard-margin case:

$$\min_{\boldsymbol{w},\rho} \left(\frac{1}{2} \|\boldsymbol{w}\|^2 - \rho \right) \quad \text{s.t.} \quad \forall i : \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i \geq \rho \ . \tag{4.23}$$

The corresponding primal Lagrangian is

$$L(\boldsymbol{w}, \rho, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{w}\|^2 - \rho - \sum_i \alpha_i (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i - \rho) \ , \tag{4.24}$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. Differentiating with respect to the primal variables yields

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_i \alpha_i \boldsymbol{x}_i = 0 \quad \Leftrightarrow \quad \boldsymbol{w} = \sum_i \alpha_i \boldsymbol{x}_i \text{ and} \tag{4.25}$$

$$\frac{\partial L}{\partial \rho} = -1 + \sum_i \alpha_i = 0 \quad \Leftrightarrow \quad \sum_i \alpha_i = 1 \ . \tag{4.26}$$

By resubstituting 4.25 and 4.26 into 4.24 and rearranging we obtain the dual

$$\min_{\boldsymbol{\alpha}} \left(\frac{1}{2} \sum_i \alpha_i \alpha_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \right) \quad \text{s.t.} \quad \sum_i \alpha_i = 1 \ , \ \forall i : \alpha_i \geq 0 \ . \tag{4.27}$$

Moreover, we can set the margin to a constant value such as $\rho = 1$, which yields the dual

$$\min_{\boldsymbol{\alpha}} \left(\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j - \sum_i \alpha_i \right) \quad \text{s.t.} \quad \forall i : \alpha_i \geq 0 \ . \tag{4.28}$$

Obviously, solving 4.27 or 4.28 yields the same maximum margin hyperplane.

Like many maximum-margin methods based on support vectors the primal 4.23 and the corresponding duals 4.27 and 4.28 implicitly maximise the margin by minimising the length of $\boldsymbol{w}$, while ensuring correct classification of all data samples by linear constraints $\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i \geq \rho$.

We, however, consider a different strategy for obtaining the maximum-margin support-vector solution—instead of minimising the length of $\boldsymbol{w}$ we explicitly maximise the margin ρ while the length of $\boldsymbol{w}$ is constant, that is

$$\max_{\boldsymbol{w}} \rho(\boldsymbol{w}) \quad \text{s.t.} \quad \|\boldsymbol{w}\| = 1 \ , \ \text{with} \tag{4.29}$$

$$\rho(\boldsymbol{w}) = \min_{\boldsymbol{x}_i} (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i) \ , \tag{4.30}$$

$$\boldsymbol{w} = \sum_i \alpha_i \boldsymbol{x}_i \ , \quad \alpha_i \geq 0 \ . \tag{4.31}$$

Geometrically, we consider the problem of finding the hyperplane $\boldsymbol{w}_*$ passing through the origin and having maximum margin ρ_* with respect to the given data samples (see Figure 4.8). Correct classification of all data samples is directly achieved by 4.30 and we assume with 4.31 that $\boldsymbol{w}$ is a linear combination of the data samples $\boldsymbol{x}_i$. Hence, Problems 4.23, 4.28 and 4.29 yield the same maximum margin hyperplane.

The decision function then becomes

$$f(\boldsymbol{x}) = \operatorname{sgn}(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} - 1) \ . \tag{4.32}$$

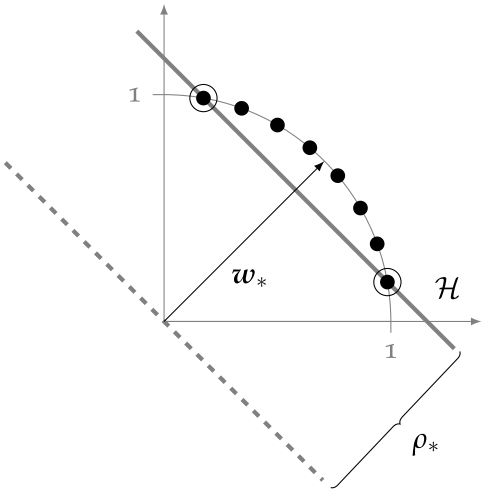


Figure 4.8.: Comparison of solutions obtained by solving problem 4.23, depicted by $\mathcal{H}$, and by solving problem 4.29 with w_* and ρ_*.

We propose a straightforward and incremental algorithm for solving Problem 4.29, where the weights α_i used in the description of the weight vector w, see Equation 4.31, are updated in each iteration according to a particular learning rule. Let w_t be the approximation of the optimal w_* at time t and ρ_t the approximation of the maximum margin ρ_*. During the iterative optimisation the constraint $\|w\| = 1$ is dropped, which is not critical as we will see later. The algorithm starts with $w_0 = \mathbf{0}$ and after $t_{\max}$ learning iterations the norm of the final approximation $w_{t_{\max}}$ is set to one. The basic idea of the novel learning algorithm is as follows. In each learning iteration the sample that is closest to the current hyperplane defined by w_t is selected, that is

$$x_{\min}(t) = \arg\min_{x_i} \left(w_t^{\mathrm{T}} x_i\right) \quad . \tag{4.33}$$

For each given training sample x_i the counter variable α_i is increased by some positive a whenever the sample is selected as $x_{\min}(t)$:

$$\alpha_i = \alpha_i + a \quad \text{for} \quad x_{\min}(t) = x_i \quad . \tag{4.34}$$

Let $\mathcal{X}'(t)$ denote the set of samples x_j for which $\alpha_j > 0$ holds at time t. Out of this set, the algorithm selects the sample being most distant to the current hyperplane defined by w_t:

$$x_{\max}(t) = \arg\max_{x_j \in \mathcal{X}'(t)} \left(w_t^{\mathrm{T}} x_j\right) \quad . \tag{4.35}$$

Whenever a sample is selected as $x_{\max}(t)$, its associated counter variable is decreased by some positive b:

$$\alpha_i = \alpha_i - b \quad \text{for} \quad x_{\max}(t) = x_i \quad . \tag{4.36}$$

Note that 4.34 and 4.36 can be combined to the learning rule

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + a\,\boldsymbol{x}_{\min}(t) - b\,\boldsymbol{x}_{\max}(t) \ , \tag{4.37}$$

with $a > b > 0$. Since the learning rule 4.37 increases the weight for samples close to the hyperplane and decreases the weight for samples farther from the hyperplane, it is still very similar to the well-known perceptron algorithm [86]. Reasonable values for the learning rate a and for the forgetting rate b will be discussed later.

The distance d of a sample $\boldsymbol{x}_j$ to the hyperplane at time t can be recovered by

$$d_j := d(\boldsymbol{x}_j) = \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_j = \sum_i \alpha_i \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \ . \tag{4.38}$$

As mentioned in the beginning, we require that the dataset has been mapped into some feature space where all samples can be linearly separated from the origin. This transformation, however, is not required explicitly; it can be achieved by replacing the standard dot product with a kernel that implements an implicit mapping to a feature space and that satisfies Mercer's conditions. Using a kernel function we can replace Equation 4.38 with

$$d(\boldsymbol{x}_j) = \sum_i \alpha_i\, k(\boldsymbol{x}_i, \boldsymbol{x}_j) \ , \tag{4.39}$$

where k is an appropriate kernel function such as the Gaussian kernel. We can further rewrite the distances to the hyperplane using the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{N \times N}$ with $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ such that

$$\boldsymbol{d} = (d_1, \ldots, d_N)^{\mathrm{T}} = \boldsymbol{K\alpha} \ . \tag{4.40}$$

Altogether, we obtain Algorithm 4.1 for simple and incremental support vector data description. The correct choice of the kernel and its parameters, such as σ in case of a Gaussian kernel, is crucial and strongly influences the shape of the decision boundary, the number of support vectors, and the resulting performance; Figure 4.9 shows solutions computed with different values of σ. A common technique for determining appropriate kernel parameters is cross validation—more details on cross validation and other techniques for model selection can be found in [44, Chapter 7], for instance.

4.3.1. Proof of Convergence

In the previous section, we have presented a novel approach for solving the optimisation problem 4.29; now we analyse our algorithm theoretically and prove (i) that the algorithm converges to the maximum margin hyperplane and (ii) that this hyperplane is solely described by support vectors. However, we will first present a few propositions that are required for the final proofs.

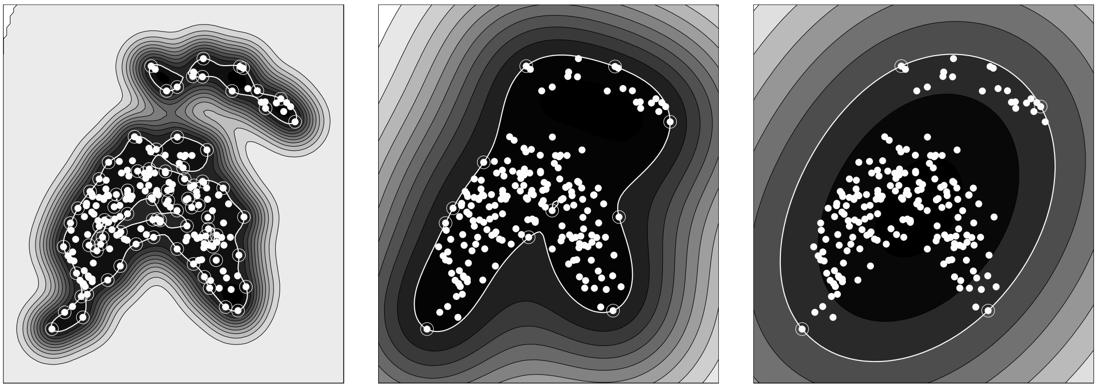
```
$\boldsymbol{\alpha} = \texttt{ommo}(\boldsymbol{K}, t_{\max}, a, b)$

Input:   $\boldsymbol{K}$        kernel matrix of data samples $\boldsymbol{x}_i$, $i = \{1, \dots, N\}$
         $t_{\max}$     number of iterations, e.g. $t_{\max} = 10^4$
         $a$        learning rate, e.g. $a = 2$
         $b$        forgetting rate, e.g. $b = 1$

Output:  $\boldsymbol{\alpha}$        coefficient vector

$\alpha_i \leftarrow 0 \quad \forall i = 1, \dots, N$
for $t \leftarrow 1, \dots, t_{\max}$ do
    $\boldsymbol{d} \leftarrow \boldsymbol{K}\boldsymbol{\alpha}$                                  ▷ compute distances
    $\boldsymbol{x}_{\min}(t) \leftarrow \arg\min_i d_i$                        ▷ determine closest sample
    $\boldsymbol{x}_{\max}(t) \leftarrow \arg\max_{i,\, \alpha_i > 0} d_i$              ▷ determine farthest support vector
    $\alpha_{\min} \leftarrow \alpha_{\min} + a$                          ▷ increase weight for closest sample
    $\alpha_{\max} \leftarrow \alpha_{\max} - b$                            ▷ decrease weight for farthest sv
end for
$\boldsymbol{d} \leftarrow \boldsymbol{K}\boldsymbol{\alpha}, \quad \rho \leftarrow \min_i d_i, \quad \boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha}/\rho$                   ▷ scale $\boldsymbol{\alpha}$ such that $\rho = 1$
```

Algorithm 4.1 (OMMO): Computes the solution of the primal Problem 4.29 using OMMO.



(a) small σ (b) intermediate σ (c) large σ

Figure 4.9.: Example toy data and the solutions of the OMMO algorithm with a Gaussian kernel. For small σ the decision boundary is tight and sensitive due to the large number of support vectors (left), whereas a large σ results in a smooth and loose decision boundary with few support vectors (right). A good decision boundary with a limited number of support vectors is obtained for some intermediate σ (middle).

Proposition 4.3.1. *The length of $\boldsymbol{w}_t$ is bounded such that $\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t)/\rho_* \leq \|\boldsymbol{w}_t\|$.*

Proof. From the definition of the margin, see Equation 4.30, we already know that the margin at time t can be obtained by $\rho_t = \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t)/\|\boldsymbol{w}_t\|$ and is upper bounded such that $\rho_t \leq \rho_*$; this directly leads to $\|\boldsymbol{w}_t\| \geq \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t)/\rho_*$. □

Proposition 4.3.2. *For $a > 0$, $b > 0$, and $a = b + 1$ the length of $\boldsymbol{w}_t$ is bounded such that $\|\boldsymbol{w}_t\| \leq \rho_*\, t + (a+b)\,\sqrt{t}$.*

Proof. This is done by induction and using the properties

$$\forall i : \|\boldsymbol{x}_i\| = 1 \ , \tag{4.41}$$

$$\forall t : \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t) \leq \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\max}(t) \ , \tag{4.42}$$

$$\forall t : \boldsymbol{x}_{\min}(t)^{\mathrm{T}}\boldsymbol{x}_{\max}(t) = \cos\beta\ \|\boldsymbol{x}_{\min}(t)\|\ \|\boldsymbol{x}_{\max}(t)\| \geq -1 \ . \tag{4.43}$$

The case $t = 0$ is trivial and for $t \to t+1$ it follows that

$$\begin{aligned}
\|\boldsymbol{w}_{t+1}\|^2 &\overset{4.37}{=} (\boldsymbol{w}_t + (a\boldsymbol{x}_{\min}(t) - b\boldsymbol{x}_{\max}(t)))^2 \\
&= \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\boldsymbol{w}_t^{\mathrm{T}}\left(a\boldsymbol{x}_{\min}(t) - b\boldsymbol{x}_{\max}(t)\right) + \left(a\boldsymbol{x}_{\min}(t) - b\boldsymbol{x}_{\max}(t)\right)^2 \\
&\overset{(a=b+1)}{=} \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t) + 2\left(\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t) - \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\max}(t)\right) \\
&\quad + a^2\|\boldsymbol{x}_{\min}(t)\| + b^2\|\boldsymbol{x}_{\max}(t)\| - 2ab\boldsymbol{x}_{\min}(t)^{\mathrm{T}}\boldsymbol{x}_{\max}(t) \\
&\overset{4.41}{=} \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t) + 2\left(\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\min}(t) - \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\max}(t)\right) \\
&\quad + a^2 + b^2 - 2ab\boldsymbol{x}_{\min}(t)^{\mathrm{T}}\boldsymbol{x}_{\max}(t) \\
&\overset{4.42}{\leq} \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\boldsymbol{w}_t^{T}\boldsymbol{x}_{\min}(t) + a^2 + b^2 - 2ab\boldsymbol{x}_{\min}(t)^{\mathrm{T}}\boldsymbol{x}_{\max}(t) \\
&\overset{4.43}{\leq} \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\boldsymbol{w}_t^{T}\boldsymbol{x}_{\min}(t) + a^2 + b^2 + 2ab \\
&\overset{4.3.1}{\leq} \boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{w}_t + 2\rho_*\|\boldsymbol{w}_t\| + (a+b)^2 \\
&\leq \left(\rho_*\, t + (a+b)\sqrt{t}\right)^2 + 2\rho_*\left(\rho_*\, t + (a+b)\sqrt{t}\right) + (a+b)^2 \\
&= \rho_*^2 t^2 + 2\rho_*^2 t + 2\rho_*(t+1)(a+b)\sqrt{t} + (a+b)^2(t+1) \\
&\leq \rho_*^2(t^2 + 2t + 1) + 2\rho_*(t+1)(a+b)\sqrt{t} + (a+b)^2(t+1) \\
&\leq \left(\rho_*\,(t+1) + (a+b)\sqrt{t+1}\right)^2 \ .
\end{aligned}$$

□

Theorem 4.3.3. *For $t \to \infty$, $a, b > 0$, and $a = b + 1$ the angle γ_t between the optimal direction $\boldsymbol{w}_*$ and the direction $\boldsymbol{w}_t$ found by* OMMO *converges to zero, i.e.* $\lim_{t\to\infty} \gamma_t = 0$.

Proof. We need the property of learning rule 4.37 that a sample can only be forgotten, if it has been learnt before, which is

$$\forall\, \boldsymbol{x}_{\max}(t)\ \exists\, \boldsymbol{x}_{\min}(t')\,,\ t' < t\ :\ \boldsymbol{x}_{\max}(t) = \boldsymbol{x}_{\min}(t')\ . \tag{4.44}$$

According to the maximum margin property, it holds that

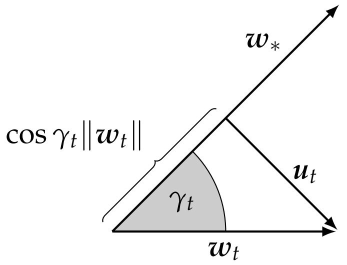
$$\boldsymbol{w}_*^{\mathrm{T}} \boldsymbol{x}_{\min}(i) \geq \rho_*\ . \tag{4.45}$$

Then the cosine of the angle between $\boldsymbol{w}_*$ and $\boldsymbol{w}_t$ can be written as:

$$\begin{aligned}
\cos\gamma_t &= \frac{\boldsymbol{w}_*^{\mathrm{T}} \boldsymbol{w}_t}{\|\boldsymbol{w}_t\|} \\
&= \frac{1}{\|\boldsymbol{w}_t\|} \sum_{i=0}^{t-1} \boldsymbol{w}_*^{\mathrm{T}} \left(a\, \boldsymbol{x}_{\min}(i) - b\, \boldsymbol{x}_{\max}(i)\right) \\
&\overset{a=b+1}{=} \frac{1}{\|\boldsymbol{w}_t\|} \sum_{i=0}^{t-1} \boldsymbol{w}_*^{\mathrm{T}} \left(\boldsymbol{x}_{\min}(i) + b\, \boldsymbol{x}_{\min}(i) - b\, \boldsymbol{x}_{\max}(i)\right) \\
&\overset{4.44}{=} \frac{1}{\|\boldsymbol{w}_t\|} \sum_{i=0}^{t-1} \boldsymbol{w}_*^{\mathrm{T}} \boldsymbol{x}_{\min}(i) \\
&\overset{4.45}{\geq} \frac{1}{\|\boldsymbol{w}_t\|} \rho_*\, t \\
&\overset{4.3.2}{\geq} \frac{\rho_*\, t}{\rho_*\, t + (a+b)\sqrt{t}} \\
&= \frac{1}{1 + \frac{(a+b)\sqrt{t}}{\rho_*\, t}} \\
&\geq 1 - \frac{a+b}{\rho_* \sqrt{t}} \\
&\overset{t\to\infty}{\longrightarrow} 1
\end{aligned}$$

□

We have thus proven that the OMMO algorithm converges to the optimal maximum-margin hyperplane with a convergence rate of at least $\mathcal{O}(1/\sqrt{t})$; now, we prove that the hyperplane computed with OMMO is solely determined by support vectors.



$$\boldsymbol{w}_t = \cos\gamma_t\|\boldsymbol{w}_t\|\boldsymbol{w}_* + \boldsymbol{u}_t \tag{4.46}$$

$$\|\boldsymbol{u}_t\| = \|\boldsymbol{w}_t\|\sin\gamma_t \tag{4.47}$$

$$\|\boldsymbol{w}_*\| = 1$$

Figure 4.10.: Orthogonal decomposition of $\boldsymbol{w}_t$ and properties that hold within this decomposition.

Theorem 4.3.4. *Beyond some finite number of iterations $t > t'$ the set $\mathcal{X}'(t)$ will always consist of support vectors only.*

Proof. We, first, show that after some finite number of iterations $t' < t$, $\boldsymbol{x}_{\text{min}}(t)$ will always be a support vector. Therefore, we use the orthogonal decomposition of $\boldsymbol{w}_t$ depicted in Figure 4.10. Furthermore, $\mathcal{X}^{\text{sv}}$ will denote the set of final support vectors.

We perform an indirect proof by assuming that a finite number of iterations t' for which $\boldsymbol{x}_{\text{min}}(t)$ with $t > t'$ will always be a support vector does not exist, i.e. $\nexists\, t' < \infty\,,\ \forall\, t > t' :\ \boldsymbol{x}_{\text{min}}(t) \in \mathcal{X}^{\text{sv}}$. Furthermore, we use the following property

$$\forall \boldsymbol{z} :\ \boldsymbol{z}^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t) = \cos(\alpha)\,\|\boldsymbol{z}\|\,\|\boldsymbol{x}_{\text{min}}(t)\| \overset{4.41}{=} \cos(\alpha)\,\|\boldsymbol{z}\| \quad \Leftrightarrow \quad \boldsymbol{z}^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t) \leq \|\boldsymbol{z}\|\ . \tag{4.48}$$

$$\begin{aligned}
\Rightarrow \boldsymbol{w}_*^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t) &> \rho_* && (4.49)\\
&\geq \rho_t\\
&= \frac{\boldsymbol{w}_t^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)}{\|\boldsymbol{w}_t\|}\\
&\overset{4.46}{=} \frac{\left(\cos\gamma_t\|\boldsymbol{w}_t\|\boldsymbol{w}_* + \boldsymbol{u}_t\right)^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)}{\|\boldsymbol{w}_t\|}\\
&= \cos\gamma_t\boldsymbol{w}_*^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t) + \frac{\boldsymbol{u}_t^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)}{\|\boldsymbol{w}_t\|}\\
&\overset{4.47}{=} \cos\gamma_t\ \boldsymbol{w}_*^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)\ +\ \frac{\boldsymbol{u}_t^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)}{\|\boldsymbol{u}_t\|}\ \sin\gamma_t\\
&\overset{4.48}{\geq} \cos\gamma_t\ \boldsymbol{w}_*^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t)\ +\ \sin\gamma_t\\
&\xrightarrow{4.3.3} \boldsymbol{w}_*^{\mathrm{T}}\boldsymbol{x}_{\text{min}}(t) \quad \text{for } t \to \infty && (4.50)
\end{aligned}$$

Due to 4.50 there is a t' where $\boldsymbol{x}_{\min}(t)$ being a non-support vector and $t > t'$ inevitably leads to a contradiction. Hence, for $t > t'$ only support vectors are added to the set $\mathcal{X}'(t)$, i.e. there is a finite number of non-support vectors contained in the set $\mathcal{X}'(t)$. As a consequence after a finite number of iterations $t'' > t$ also $\boldsymbol{x}_{\max}(t'')$ will always be a support vector.

Finally, we prove that all non-support vectors in the set $\mathcal{X}'(t)$ will be removed. *Assumption*: There exists a sample $\boldsymbol{x}$ that is not a support vector but it remains in the set $\mathcal{X}'(t)$, i.e. $\exists\, \boldsymbol{x} : \boldsymbol{x} \notin \mathcal{X}^{\text{sv}} \wedge \boldsymbol{x} \in \mathcal{X}'(t)$ for all t.

$$\Longrightarrow \; \rho_* \; < \; \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}\boldsymbol{x} \; < \; \frac{\boldsymbol{w}_t}{\|\boldsymbol{w}_t\|}\boldsymbol{x}_{\max}(t) \; \overset{t\to\infty}{\longrightarrow} \; \rho_* \tag{4.51}$$

Since after a finite number of iterations $\boldsymbol{x}_{\max}(t)$ will always be a support vector, the assumption leads to a contradiction and hence all non-support vectors in the set $\mathcal{X}'(t)$ will be removed. □

We have proven that the OMMO algorithm converges with at least $\mathcal{O}(1/\sqrt{t})$ to the maximum margin solution that is solely described by support vectors. Our novel algorithm can thus be used, without specific knowledge in optimisation theory, for efficient data description and novelty detection. Moreover, OMMO is an alternative to existing optimisation toolboxes or sophisticated training algorithms, especially for novices in the field of machine learning.

4.3.2. Soft-Margin Kernelised OneClassMaxMinOver

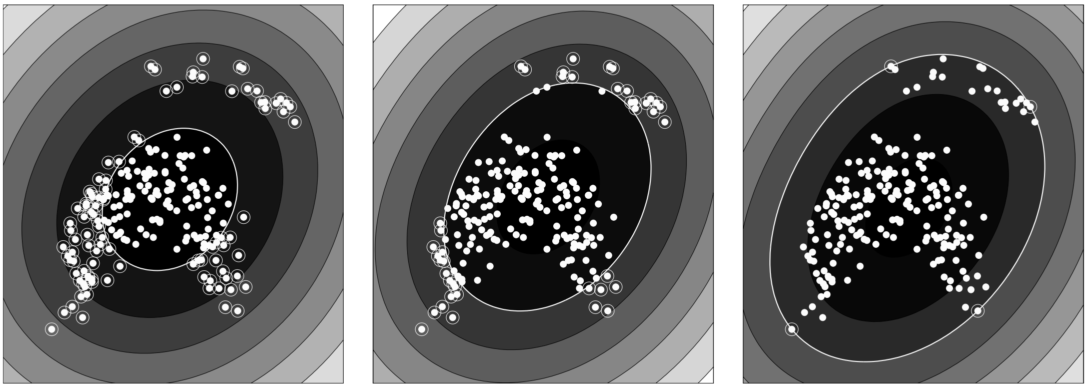
So far, we have considered the hard-margin problem without allowing a sample on the other side of the separating hyperplane. To realise a soft-margin we employ slack variables ξ_i such that the quadratic optimisation problem becomes

$$\min_{\boldsymbol{w},\xi} \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{2}\sum_i \xi_i^2 \right) \quad \text{s.t.} \quad \forall i : \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}_i) \geq 1 - \xi_i \;\; . \tag{4.52}$$

In the hard-margin case ($C \to \infty$) this is equivalent to Problem 4.29. Note that compared to the Problems 4.12 and 4.17 the non-negativity constraint on each slack variable, i.e. $\xi_i \geq 0$, disappears, since we use a 2-norm penalisation term in the objective function. By constructing the primal Lagrangian of 4.52, setting the partial differentiations to zero, and rearranging we obtain

$$\min_{\boldsymbol{\alpha}} \left(\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j \left(k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{1}{C}\delta_{ij} \right) - \sum_i \alpha_i \right) \;\; \text{s.t.} \;\; \forall i : \; \alpha_i \geq 0 \;\; , \tag{4.53}$$

where δ_{ij} is the Kronecker delta, which is 1 if $i = j$ and 0 otherwise. As mentioned in [25] this can be understood as solving the hard-margin problem in a modified kernel



(a) small C (b) intermediate C (c) large C

Figure 4.11.: Example toy data and the solutions of the OMMO algorithm that incorporates softness; we, here, applied a Gaussian kernel with large σ, compare Figure 4.9c. The parameter C controls the softness of the class boundary; for small C many data samples are outside the description (left) and as C increases the number of outliers decreases (middle); for large C the class boundary covers all data samples (right). Since C is an additional kernel parameter, appropriate values for C can be computed, for instance, by cross validation.

space with the kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{1}{C}\delta_{ij}$. To implement a 2-norm soft-margin version of OMMO we therefore apply a simple modification to Algorithm 4.1 such that

$$d(\boldsymbol{x}_i) = \sum_j \alpha_j \left(k(\boldsymbol{x}_j, \boldsymbol{x}_i) + \frac{1}{C}\delta_{ij} \right) \quad \text{and} \tag{4.54}$$

$$\boldsymbol{d} = \boldsymbol{K}^* \boldsymbol{\alpha} \quad \text{with} \tag{4.55}$$

$$\boldsymbol{K}^* = \boldsymbol{K} + \frac{1}{C}\boldsymbol{I} \quad , \tag{4.56}$$

where $\boldsymbol{I}$ is the identity matrix. Figure 4.11 shows the results of the modified OMMO for different C and a Gaussian kernel with large σ.

4.3.3. Optimisations

Since OMMO is simple and can be implemented within only a few lines of source code and without any sophisticated toolboxes, we can easily integrate several modifications and optimisations. In the following, we present the most important optimisations:

Stopping Criteria Several ways for defining stopping criteria of an iterative support vector approach such as OMMO have been proposed in the literature, e.g. monitoring the growth of the dual objective function, monitoring the Karush-Kuhn-Tucker complementary conditions for the primal, or measuring the feasibility gap. For a detailed discussion on stopping criteria see [25, chapter 7] or [90, chapter 10]. We, here, focus on the so-called feasibility gap f, which is defined as the difference between the values of the primal 4.52 and the dual objective function 4.53:

$$\begin{aligned} f(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{K}) &= \text{primal} - \text{dual} && (4.57) \\ &= \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \frac{C}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} - \boldsymbol{e}^{\mathrm{T}}\boldsymbol{\alpha} + \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{K}^{*}\boldsymbol{\alpha} && (4.58) \\ &= \boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{d} - \frac{1}{2C}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\alpha} + \frac{C}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} - \boldsymbol{e}^{\mathrm{T}}\boldsymbol{\alpha} \ , && (4.59) \end{aligned}$$

where $\boldsymbol{e} = (1, \ldots, 1)^{\mathrm{T}}$ and $\boldsymbol{d} = (d_1, \ldots, d_N)^{\mathrm{T}} = \boldsymbol{K}^{*}\boldsymbol{\alpha} = \boldsymbol{K}\boldsymbol{\alpha} + \frac{1}{C}\boldsymbol{\alpha}$. A useful measure of progress is the normalised feasibility gap

$$\begin{aligned} f^{*}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{K}) &= \frac{\text{primal} - \text{dual}}{\text{primal} + 1} && (4.60) \\ &= \frac{2\boldsymbol{\alpha}^{\mathrm{T}} - \frac{1}{C}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\alpha} + C\,\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} - 2\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\alpha}}{\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{d} + C\,\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} + 2} \ . && (4.61) \end{aligned}$$

Since we apply a 2-norm slack term and use a modified kernel function, we must compute the correct ξ_i to obtain f^{*}; this is done by normalising α_i such that the distance of the hyperplane to the origin equals one:

$$\alpha_i \leftarrow \frac{\alpha_i}{\rho} \quad , \quad \text{with} \quad \rho = \min_i d_i \ , \qquad (4.62)$$

where d_i is the distance of sample $\boldsymbol{x}_i$ to the current hyperplane, see Equation 4.39. The evaluation of f^{*} takes time $\mathcal{O}(N)$ due to the dot products, however, it is sufficient to compute f^{*} in regular intervals, e.g. every 100th step; hence, the runtime of OMMO does not change significantly.

Kernel Caching The kernel evaluations d_i are usually the most time consuming computations of naïve implementations. Whereas for some applications the full kernel matrix can be computed beforehand, especially in large sample size scenarios kernel values must be computed online. Instead of recomputing identical kernel values we apply a kernel cache that stores frequently used values for later usage. The size of the cache is crucial depending on the number of support vectors of the final solution, but in most cases only few data samples become support vectors and, hence, the cache often contains more elements than final support vectors.

Incremental Kernel Evaluation Learning rule 4.1 of OMMO changes only $\alpha_{\max}$ and $\alpha_{\min}$ during each iteration. Hence, d_i can be evaluated incrementally by

$$d_i(t+1) = d_i(t) + 2\,k(\boldsymbol{x}_{\min}(t), \boldsymbol{x}_i) - k(\boldsymbol{x}_{\max}(t), \boldsymbol{x}_i) \ . \qquad (4.63)$$

As this incremental summation can be numerically instable, s_i should be recomputed after a fixed number of iterations, for instance after 1000 iterations. The additional cost in complexity can be neglected, especially when this recomputation is performed in combination with the stopping criterion.

Preinitialisation With grid search we can evaluate the performance of a machine learning algorithm for different parameters, such as the kernel parameter σ in case of a Gaussian kernel and the softness parameter C. For most applications these parameters are sampled on a regular grid and the optimal parameters are chosen according to common performance measures such as the area under the receiver-operator-characteristic or cross-validation error. Since parameters of adjacent nodes in the grid will yield similar support vector solutions, we can initialise OMMO with the solution of an adjacent node. In this case, we must omit the scaling of α_i, compare last line of Algorithm 4.1. Moreover, if a transformation between kernel values with different parameters exists and if it requires less time than a complete recalculation, the kernel values can also be reused. For Gaussian kernels,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) \leftarrow \left(k(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{\delta_{ij}}{C_1} \right)^{\sigma_1^2/\sigma_2^2} + \frac{\delta_{ij}}{C_2} \tag{4.64}$$

transforms the kernel values of the parameter tuple (C_1, σ_1) into those of the tuple (C_2, σ_2).

Algorithm 4.2 shows a modified version of OMMO that employs the evaluation of the feasibility gap and incremental kernel evaluation; this requires only three additional lines in the loop. We will demonstrate later that preinitialisation in combination with the normalised feasibility gap as stopping criterion and the incremental kernel evaluation yield a significant speed-up when performing cross validation.

4.4. Experiments and Results

In this section, we analyse the behaviour of OMMO for different parameters and with the aforementioned optimisations. Moreover, we compare with a state-of-the-art algorithm for solving Problem 4.52; if not otherwise noted, we use positive samples from the *banana* benchmark dataset[1] (see Figure 4.12), a Gaussian kernel with parameter σ and softness parameter C, and learning rates $a = 2$ and $b = 1$.

4.4.1. Feasibility Gap and Convergence Rate

Instead of applying OMMO with a fixed number of iterations, the normalised feasibility gap, Equation 4.60, can be used as accuracy measure during the incremental

[1] `http://archive.ics.uci.edu/ml`

```
$\boldsymbol{\alpha} = \texttt{ommo2}(\boldsymbol{K}, f_{\text{stop}}, a, b, C)$

Input:   $\boldsymbol{K}$           kernel matrix with $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $i, j \in \{1, \ldots, N\}$
         $f_{\text{stop}}$    stopping criterion for the feasibility gap, e.g. $10^{-3}$
         $a$           learning rate
         $b$           forgetting rate
         $C$           softness parameter

Output:  $\boldsymbol{\alpha}$           coefficient vector

$\alpha_i \leftarrow 0 \quad \forall i = 1, ..., N$                                  ▷ initialise counter variables
$f \leftarrow \infty$                                                 ▷ initialise feasibility gap
$\boldsymbol{K}^* \leftarrow \boldsymbol{K} + \frac{1}{C}\boldsymbol{I}$                                          ▷ modified kernel matrix
$\boldsymbol{d} \leftarrow \boldsymbol{K}^*\boldsymbol{\alpha}$                                               ▷ compute distances
while $f > f_{\text{stop}}$ do
    $\boldsymbol{x}_{\min}(t) \leftarrow \arg\min_i d_i$                             ▷ determine closest sample
    $\boldsymbol{x}_{\max}(t) \leftarrow \arg\max_{i,\, \alpha_i > 0} d_i$                    ▷ determine farthest support vector
    $\alpha_{\min} \leftarrow \alpha_{\min} + a$                               ▷ increase weight for closest sample
    $\alpha_{\max} \leftarrow \alpha_{\max} - b$                               ▷ decrease weight for farthest sv
    $\boldsymbol{d} \leftarrow \boldsymbol{d} + a\,\boldsymbol{K}^*_{.,\min} - b\,\boldsymbol{K}^*_{.,\max}$                  ▷ update distances incrementally
    $\rho \leftarrow \min_i d_i$                                   ▷ determine scaling parameter
    $\boldsymbol{\xi} \leftarrow \max\left(\boldsymbol{0}, \boldsymbol{1} - \left(\boldsymbol{d} - \boldsymbol{\alpha}/C\right)/\rho\right)$                  ▷ compute slack variables
    $f \leftarrow \dfrac{\frac{2}{\rho^2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{d} - \frac{1}{C\rho^2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\alpha} + C\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} - \frac{2}{\rho}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\alpha}}{\frac{1}{\rho^2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{d} + C\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} + 2}$      ▷ current feasibility gap
end while
$\boldsymbol{d} \leftarrow \boldsymbol{K}^*\boldsymbol{\alpha} \quad , \rho \leftarrow \min_i d_i \quad , \boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha}/\rho$         ▷ scale $\boldsymbol{\alpha}$ such that margin$= 1$
```

Algorithm 4.2 (OMMO2): Computes the support vector solution using OMMO with stopping criterion and incremental kernel evaluation.

computation, see Algorithm 4.2. This, however, does not reduce the number of parameters, since we have to select a threshold for the feasibility gap. We, here, address the question whether the number of iterations for different kernel parameters changes significantly when using a fixed feasibility gap as stopping criterion for OMMO. We randomly selected 200 data samples from the banana dataset and we stopped OMMO when the normalised feasibility gap has reached the value 10^{-4}. The results in Table 4.1 demonstrate that the number of iterations significantly increases as C increases and if σ

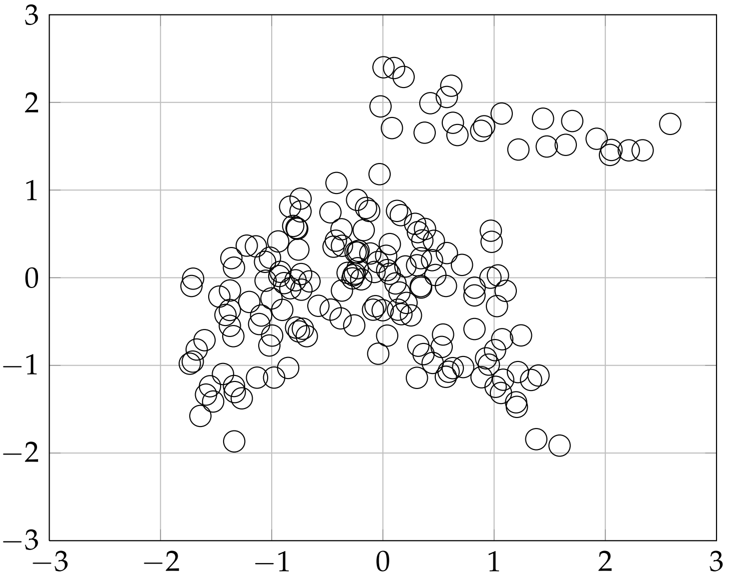


Figure 4.12.: Positive data samples of the banana dataset used throughout the experiments if not otherwise noted; the data samples have zero mean and unit variance.

is constant; for a constant C the number of iterations decreases as σ increases. Since the number of iterations is generally proportional to the number of final support vectors, these results are reasonable and show that there is no optimal strategy for choosing the number of iterations that OMMO should perform. Hence, the feasibility gap is not only an appropriate measure for controlling OMMO, but will also reduce computation time.

Theoretically, we have proven that OMMO converges with at least $\mathcal{O}(1/\sqrt{t})$ to the maximum margin solution that is solely based on support vectors. Here, we will have a closer look at the convergence rate and the behaviour of OMMO. We, therefore, use some intermediate kernel values, i.e. $\sigma = 0.8$ and $C = 5$, and compute the normalised feasibility gap during each iteration of OMMO. Figure 4.13 indicates even faster convergence than we have proven; within the first ten iterations the feasibility gap is almost constant, but thereafter convergence is really fast—for 10^3 iterations OMMO achieves a feasibility gap of less than 10^{-3} and after 10^5 iterations the feasibility gap is approximately 10^{-7}. Moreover, Figure 4.14 illustrates the behaviour of OMMO when the number of iterations varies. Whereas for small t the class boundary describes the data samples roughly, for $t > 100$ a tight description of the data samples is already visible; as $t > 10^3$ increases the solution does not change qualitatively.

Although this experiment was only conducted with a particular dataset and a particular set of kernel parameters, it shows very fast convergence and that, at least in some cases, one can expect a faster convergence rate than we have proven.

Table 4.1.: Number of iterations for OMMO required to reach a feasibility gap of 10^{-4} for 200 randomly selected samples of the banana dataset. For small σ and large C almost 600000 iterations have to be performed, whereas less than 8000 iterations are required for large σ. For intermediate σ and C the number of iterations vary between 10000 and 100000.

σ	$C = 0$	1	2	5	10	100	1000	10000
0.10	18373	20594	23847	29374	36631	82663	226028	579872
0.25	17339	18373	19608	21270	23568	37889	83687	177325
0.50	15694	16239	16780	17459	17761	21731	38527	70759
0.75	14194	14603	15021	15529	15864	17865	24629	32532
1.00	13105	13431	13782	14190	14520	15728	19985	26370
1.50	11178	11547	11906	12331	12563	12327	14123	19258
2.00	9939	10248	10561	10884	11117	10354	11626	13197
5.00	7280	7427	7536	7424	7441	7448	7429	7523

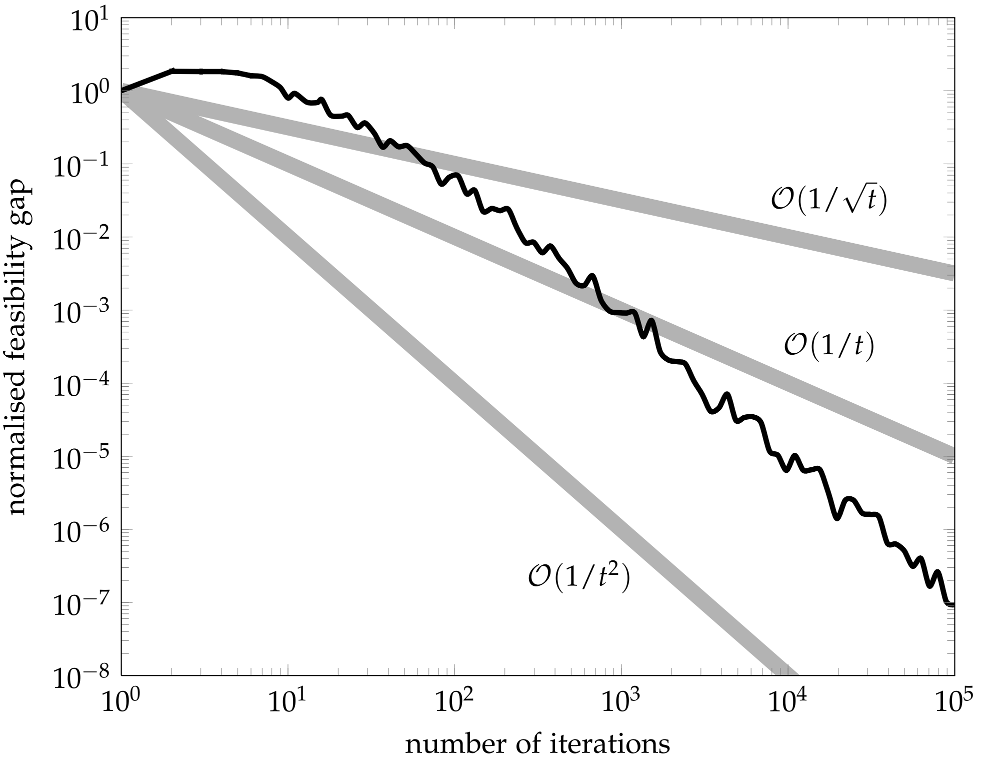
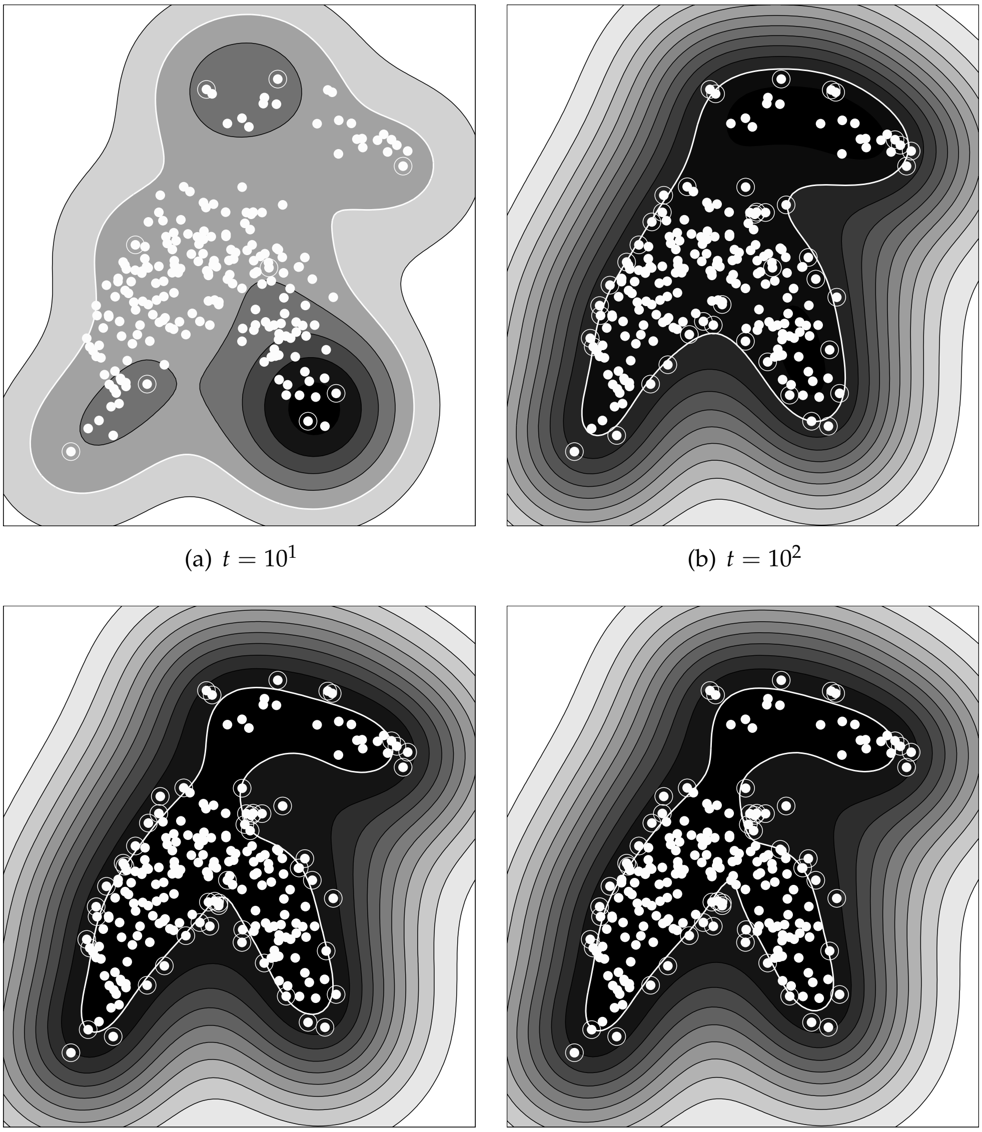


Figure 4.13.: Normalised feasibility gap (see Equation 4.60) evaluated during each iteration of OMMO. Whereas we have proven a convergence rate of at least $\mathcal{O}(1/\sqrt{t})$, we can experimentally identify a convergence rate of even $\mathcal{O}(1/t^2)$.



(a) $t = 10^1$

(b) $t = 10^2$

(c) $t = 10^3$

(d) $t = 10^5$

Figure 4.14.: Solutions of OMMO at different iterations t and with a Gaussian kernel where $\sigma = 0.8$ and $C = 5$. After 10 iterations the class boundary is very loose with few support vectors and the data samples are described imprecisely (a). In contrast, for $t = 10^2$ the class boundary has evolved into a tighter data description, where most of the true support vectors have been identified (b). After 10^3 iterations the solution is refined such that the description becomes tighter around the data samples and the remaining true support vectors have been detected (c). Finally, the shape of the boundary does not change significantly with additional iterations (d).

4.4.2. Stability Analysis

Since in each iteration of OMMO the coefficient vector $\boldsymbol{\alpha}$ remains unscaled such that $\|\boldsymbol{w}\| \neq 1$, the length of $\boldsymbol{\alpha}$ will increase constantly; this can lead to numerical issues, especially when computing d_i—compare incremental kernel evaluation 4.39. However, for most applications these issues can be neglected, since $\|\boldsymbol{\alpha}\|$ and d_i are upper bounded.

Theorem 4.4.1. *$d_i(t)$ is upper bounded by t.*

Proof. We assume that a kernel function is applied for which $k(\boldsymbol{x}, \boldsymbol{z}) \leq 1$ holds for all $(\boldsymbol{x}, \boldsymbol{z})$—for example a Gaussian. Hence,

$$d_i(t) = \sum_{j=1}^{N} \alpha_j(t)\, k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \leq \sum_{j=1}^{N} \alpha_j(t) \leq t \ . \tag{4.65}$$

We applied Equation 4.44, which states that a sample can only be forgotten, if it has been learnt before. Moreover, d_i can reach the upper bound only if the same α_j is increased and $a = b + 1, b > 0$. Then,

$$\forall t > 0 : \quad \boldsymbol{x}_{\max}(t) = \boldsymbol{x}_{\max}(t-1) \tag{4.66}$$

$$\Rightarrow \quad \boldsymbol{x}_{\max}(t) = \boldsymbol{x}_{\min}(t) \tag{4.67}$$

$$\Rightarrow \quad \alpha_{\max} \leftarrow \alpha_{\max} + 1 \ , \tag{4.68}$$

which leads to a contradiction, since a hyperplane is always described by at least two data samples. As a consequence, $d_i(t)$ is always below t. □

Since d_i is bounded by the number of iterations t, which is itself limited in real applications, compare Table 4.1, especially when applying the feasibility gap as stopping criterion, the d_i will be in a numerically stable range. However, the incremental kernel evaluation of d_i, i.e. Equation 4.63, can be numerically unstable, if $a\, k(\boldsymbol{x}_i, \boldsymbol{x}_{\max}(t)) \approx b\, k(\boldsymbol{x}_i, \boldsymbol{x}_{\min}(t))$ holds for all $\boldsymbol{x}_i$, which then would not change the $d_i(t)$. In practice, this property holds only for few $\boldsymbol{x}_i$ and an extremely large sample size.

4.4.3. Computation Time

We compare the computation time of OMMO and LIBSVM [15], which is a state-of-the-art toolbox for computing support vector solutions and which has been successfully applied in several competitions as well as real-world applications. Various optimisation techniques for fast support vector learning have been integrated into LIBSVM; the current release 3.1 employs a sequential minimisation optimisation (SMO) like algorithm with kernel caching and sophisticated working set selection as described in [34].

We have used various benchmark datasets from the UCI repository[2]; we scaled the

[2] http://archive.ics.uci.edu/ml

data samples to zero mean and unit variance, and we applied a Gaussian kernel with $\sigma = 1.5$ for the experiments. Since the softness parameters of OMMO and LIBSVM are difficult to compare, we have used extreme values such that a hard margin solution is obtained, i.e. $C = 10^6$ and $\nu = 10^{-3}$. For OMMO's learning rule we have chosen $a = 2$, $b = 1$, and we have stopped the algorithm once a feasibility gap of 10^{-4}, computed within each iteration, has been reached. For LIBSVM we set the stopping criterion to 10^{-4}, which yields approximately the same feasibility gap.

Table 4.2 shows the runtime comparison of both methods; the number of iterations OMMO performs to reach the feasibility gap varies between 400 and 5500, which demonstrates very fast convergence; regarding the number of support vectors both methods yield comparable results for almost all datasets, which indicates, in combination with the parameter setting, that the decision boundaries are also comparable; only for the datasets *flare-solar* and *titanic* OMMO obtains considerably more support vectors, which can not be validated since these datasets are high-dimensional; it is, therefore, hard to identify the true support vectors. Since for LIBSVM a threshold for determining the support vectors must be chosen, we believe that this value may lead to numerical issues and strongly influence the number of support vectors for the mentioned datasets; in contrast, the support vectors can be obtained simply and without any numerical issues, if we set $a = b + 1$ and $b \in \mathbb{N} \setminus 0$ for the OMMO algorithm.

The computation time of LIBSVM varies between 0.1 ms and 19.0 ms, whereas the computation time of OMMO only varies between 0.5 ms and 10.0 ms. The LIBSVM approach obtains faster solutions for 16 datasets, whereas the OMMO approach requires less computation time for 10 datasets. In case of high-dimensional datasets such as *ringnorm* or *twonorm* OMMO significantly outperforms LIBSVM by a factor of 3 to 6.

We can summarise that in most cases the computation times of OMMO and LIBSVM are comparable and that only minor differences arise due to different implementation details.

4.4.4. Performance on UCI Benchmark Database

In this section, we compare the classification performance of all novelty-detection methods we have described so far—these are OMMO, LIBSVM, KDE, and KPCA. We use the same benchmark datasets as we have used for comparing computation times. Since we want to estimate the generalisation error, we split each dataset into 100 train and test sets (roughly 60% : 40%) and we determined the optimal parameters by performing 10-fold cross validation for the first five realisations; for reliability we take the median over the five estimates of optimal parameters, which is a standard procedure for comparing machine learning approaches on benchmark datasets, see [84] for example.

Table 4.2.: Runtime analysis of ommo and libsvm for several benchmark datasets; both methods were implemented in C/C++ and tested on an Intel Core2 2.4 GHz computer. N is the number of samples and D the number of dimensions for each dataset. We analysed the number of support vectors (SVs), the computation time t in milliseconds and, for ommo only, the required number of iterations.

Dataset	Class	N	D	libsvm		ommo		
				SVs	t [ms]	SVs	t [ms]	iter.
banana	+1	2376	2	7	**1.74**	7	3.50	1092
banana	−1	2924	2	7	**2.65**	7	4.72	937
breast-cancer	+1	81	9	8	**0.24**	8	0.55	425
breast-cancer	−1	196	9	14	**0.43**	15	0.85	589
diabetis	+1	268	8	11	**0.39**	12	2.14	1511
diabetis	−1	500	8	18	**0.82**	18	3.07	1875
flare-solar	+1	589	9	26	**1.52**	82	9.64	5470
flare-solar	−1	477	9	23	**0.89**	147	6.69	4167
german	+1	300	20	19	2.84	19	**1.56**	1069
german	−1	700	20	22	3.79	23	**2.33**	1316
heart	+1	120	13	9	**0.24**	12	0.68	508
heart	−1	150	13	11	**0.35**	12	0.85	640
image	+1	1320	18	20	**8.86**	24	7.64	3265
image	−1	990	18	23	**3.45**	26	6.60	3279
ringnorm	+1	3664	20	31	16.13	32	**5.18**	694
ringnorm	−1	3736	20	32	17.12	32	**4.05**	613
splice	+1	1527	60	40	16.54	48	**1.08**	436
splice	−1	1648	60	46	18.90	48	**1.06**	408
thyroid	+1	65	5	6	**0.12**	6	0.56	445
thyroid	−1	150	5	8	**0.18**	8	1.66	1248
titanic	+1	711	3	8	**0.61**	260	1.03	555
titanic	−1	1490	3	6	**0.83**	442	4.34	1729
twonorm	+1	3703	20	27	12.26	27	**2.88**	638
twonorm	−1	3697	20	30	13.41	30	**3.76**	768
waveform	+1	1647	21	29	7.24	29	**1.96**	691
waveform	−1	3353	21	28	12.84	27	**4.26**	849

Table 4.3 shows the results of OMMO and LIBSVM on 26 benchmark datasets. We can identify that σ, the parameter of the Gaussian kernel obtained through 10-fold cross validation, is in the same range for most datasets; only for the datasets *ringnorm*, *splice*, *thyroid*, and *twonorm* the values are slightly different. Regarding the softness parameters ν and C, the optimal values are similar except for the positive classes of the datasets *heart*, *splice*, and *twonorm*. Since we evaluated σ, ν, and C on a discretised grid with 20 nodes for each dimension (parameter), the observed differences may vanish once the resolution of the grid is increased by orders of magnitude. This assumption is mainly supported by the observation of the performance rates of both approaches; the classification performance is evaluated by computing (i) mean and standard deviation of the balanced error and (ii) the area under the receiver-operator characteristic (AUC) over the 100 test sets. Even though the optimal parameters of both approaches differ slightly for some datasets, we cannot, in almost all cases, identify significant differences, neither for the error rate nor for the AUC. Moreover, the error rate might indicate that either approach outperforms the other, see for example positive class of the dataset *splice* (31% vs. 48%), but the performance for AUC (0.74 vs. 0.77) is almost the same. Only for positive samples of the dataset *thyroid* our OMMO approach significantly outperforms LIBSVM for the error rate (44% vs. 69%) as well as for the AUC (0.51 vs. 0.26). The results, however, demonstrate that OMMO and LIBSVM yield comparable kernel parameters, which lead to similar support-vector solutions, and they thus achieve comparable results in terms of different error measures.

So far, we have found that OMMO and LIBSVM compute similar support vector solutions, now we compare both boundary-based methods with KDE and KPCA—the two density-based methods described in the beginning of this chapter. Since both density methods do not automatically provide a class boundary, we compare the four approaches only according to their AUC. Table 4.4 shows the performance regarding AUC of OMMO, LIBSVM, KPCA, and KDE for the 26 benchmark datasets; Table 4.5 shows the corresponding ranks. We can make two major observations:

Overall Performance For only 7 out of 26 datasets one particular method significantly outperforms the others; in 5 out of these 7 cases KDE yields superior performance and in 2 cases KPCA significantly outperforms the others. Especially for the positive class of the dataset *ringnorm* the KPCA method achieves the best AUC by far (0.99 vs. 0.15, 0.34, 0.03); this 20-dimensional dataset is artificially created and the analysis of all two-dimensional projections shows that negative samples are uniformly distributed in a hypersphere, which is surrounded by positive samples uniformly distributed in a 20-torus partially overlapping with the hypersphere of the negative class. Due to this special characteristic KPCA can capture the manifold of the positive class more accurately than OMMO and LIBSVM, which require many support vectors to describe the

Table 4.3.: Performance comparison of LIBSVM and OMMO for several benchmark datasets—see text for explanation.

Dataset	Class	LIBSVM					OMMO				
		σ	ν	Error [%]	Std.	AUC	σ	C	Error [%]	Std.	AUC
banana	+1	0.519	0.085	19.22	2.03	0.856	0.456	8.7E+04	19.59	1.78	0.859
banana	−1	0.519	0.010	15.68	1.66	0.910	0.688	2.1E+02	16.25	2.11	0.902
breast-c.	+1	2.696	0.004	46.31	5.34	0.556	2.366	1.5E+03	45.37	5.34	0.557
breast-c.	−1	2.194	0.024	38.86	2.89	**0.632**	1.567	3.2E+04	38.21	3.33	0.655
diabetis	+1	1.454	0.007	43.14	2.80	0.587	1.567	1.2E+04	42.82	2.78	**0.601**
diabetis	−1	1.183	0.016	35.28	1.70	0.695	1.039	1.2E+04	35.96	1.77	0.707
flare-solar	+1	0.784	0.004	50.75	6.16	0.490	0.302	5.7E+02	44.43	5.68	0.493
flare-solar	−1	0.423	0.016	43.93	5.87	0.616	0.371	2.1E+02	39.31	5.36	**0.665**
german	+1	2.696	0.007	47.86	2.80	0.537	1.926	1.2E+04	46.38	2.19	0.556
german	−1	2.696	0.010	44.33	1.35	0.587	2.366	1.2E+04	42.30	1.47	0.598
heart	+1	3.312	0.010	37.26	3.35	0.707	2.910	1.4E+00	34.36	3.49	0.696
heart	−1	2.696	0.007	32.53	3.21	0.753	3.572	1.2E+04	35.09	2.36	0.735
image	+1	1.183	0.010	17.91	1.07	0.877	1.276	8.7E+04	17.25	1.05	0.864
image	−1	1.183	0.016	18.80	1.20	0.894	1.039	8.7E+04	18.24	1.35	0.904
ringnorm	+1	0.423	0.004	50.00	0.01	**0.347**	1.276	4.3E+03	50.00	0.01	0.150
ringnorm	−1	4.070	0.003	4.66	0.82	0.998	6.624	3.2E+04	3.43	0.68	0.998
splice	+1	5.000	0.010	**31.39**	0.65	0.743	3.572	5.0E+02	48.61	0.29	0.774
splice	−1	0.638	0.004	50.00	0.03	0.348	1.276	1.2E+04	50.00	0.02	**0.414**
thyroid	+1	1.786	0.001	69.53	7.78	0.262	1.039	3.2E+04	**44.83**	6.36	**0.510**
thyroid	−1	1.786	0.004	6.01	1.76	0.988	2.366	7.6E+01	5.98	1.75	0.988
titanic	+1	0.344	0.010	51.69	10.11	0.473	0.371	5.7E+02	50.58	4.96	0.514
titanic	−1	0.344	0.016	44.97	7.76	0.606	1.039	1.2E+04	36.56	6.01	0.676
twonorm	+1	4.070	0.007	16.98	1.99	0.913	2.908	3.7E+00	20.39	1.08	0.915
twonorm	−1	4.070	0.056	17.28	1.73	0.910	2.908	4.3E+03	19.20	1.11	0.913
waveform	+1	5.000	0.004	18.55	0.91	0.899	4.389	1.4E+00	21.11	1.23	0.901
waveform	−1	2.696	0.016	37.68	2.30	0.684	2.366	4.3E+03	34.53	2.07	0.708

border of the 20-torus precisely. Likewise, positive samples of the dataset *thyroid* show similar characteristics such that KPCA yields superior performance. In contrast, KPCA fails if the dataset cannot be covered by a manifold, see for example negative samples of *ringnorm* (AUC: 0.002) or negative samples of *thyroid* (AUC: 0.031), whereas OMMO, for instance, yields accurate results (*ringnorm* 0.998 or *thyroid* 0.988). The corresponding ranks, see Table 4.5, show that KDE ranks first for many datasets, although the difference compared to the method that ranks second is insignificant, in most cases. However, KDE yields the best average rank (1.88)—slightly better compared to OMMO (2.00). Even though KPCA achieves superior performance for some datasets the overall rank is worse (3.34).

Variances in Performance The rank analysis demonstrates that KPCA has the highest performance variance; for few datasets KPCA yields superior performance by far, whereas for most of the benchmark datasets KPCA ranks last. The KDE approach shows medium variance in performance; for many datasets KDE ranks first—however, mostly not significantly better compared to the method that ranks second—but for some datasets KDE ranks last. In contrast, both support-vector methods, OMMO and LIBSVM, achieve very stable performances with small variations (standard deviation 0.69 and 0.67), which demonstrates that they can be successfully applied to a wide range of datasets.

Finally, the results demonstrate that none of the novelty-detection methods we have used here performs best for every benchmark dataset. Support-vector approaches, such as OMMO and LIBSVM, are efficient and yield accurate results for various datasets, whereas KPCA yields accurate results only for particular datasets. In contrast, KDE demonstrates accurate results for most datasets, but in few cases its performance can degenerate. Even though KDE seems to outperform OMMO and LIBSVM for some benchmark datasets, there are several disadvantages in terms of efficiency and computation time that prevent KDE to be used in large-scale applications; in these cases efficient methods such as OMMO are more appropriate.

4.4.5. Performance on Face Detection

Even though the benchmark datasets from the previous section also include real-world examples, we want to apply OMMO and LIBSVM to the problem of face detection. Recently, face detection has become very popular as it has been integrated into consumer digital cameras, for example. In its nature, face detection is a novelty-detection problem, where the target class contains images of faces and the outlier class consists of all other images. In contrast, state-of-the-art approaches treat face detection as a two-class classification problem, where non-face images are randomly sampled from a large database; this may be a valid procedure from a technical point of view, but collecting such a large dataset

Table 4.4.: Comparison of mean area under the curve (AUC) on benchmark datasets; standard deviation are given in brackets. Methods that significantly outperform all other methods for a particular dataset are drawn in boldface.

Dataset	Class	OMMO	LIBSVM	KPCA	KDE
banana	+1	0.859 (0.023)	0.856 (0.026)	0.479 (0.029)	**0.909 (0.012)**
banana	−1	0.902 (0.019)	0.910 (0.016)	0.397 (0.010)	0.925 (0.007)
breast-cancer	+1	0.557 (0.059)	0.556 (0.061)	0.503 (0.057)	0.484 (0.060)
breast-cancer	−1	0.655 (0.029)	0.632 (0.031)	0.351 (0.031)	0.652 (0.029)
diabetis	+1	**0.601 (0.033)**	0.587 (0.033)	0.521 (0.027)	0.469 (0.031)
diabetis	−1	0.707 (0.015)	0.695 (0.015)	0.268 (0.014)	0.757 (0.015)
flare-solar	+1	0.493 (0.047)	0.490 (0.040)	0.587 (0.029)	0.358 (0.015)
flare-solar	−1	0.665 (0.024)	0.616 (0.040)	0.262 (0.017)	0.685 (0.013)
german	+1	0.556 (0.030)	0.537 (0.030)	0.445 (0.026)	0.563 (0.024)
german	−1	0.598 (0.015)	0.587 (0.015)	0.382 (0.018)	0.625 (0.018)
heart	+1	0.696 (0.042)	0.707 (0.043)	0.348 (0.042)	0.746 (0.038)
heart	−1	0.735 (0.034)	0.753 (0.031)	0.235 (0.028)	**0.807 (0.024)**
image	+1	0.864 (0.007)	0.877 (0.008)	0.231 (0.010)	**0.941 (0.007)**
image	−1	0.904 (0.008)	0.894 (0.008)	0.225 (0.016)	0.896 (0.005)
ringnorm	+1	0.150 (0.004)	0.347 (0.050)	**0.993 (0.001)**	0.029 (0.007)
ringnorm	−1	0.998 (0.000)	0.998 (0.000)	0.002 (0.000)	0.999 (0.000)
splice	+1	0.774 (0.005)	0.743 (0.006)	0.350 (0.069)	**0.815 (0.007)**
splice	−1	0.414 (0.009)	0.348 (0.011)	0.589 (0.006)	0.412 (0.008)
thyroid	+1	0.510 (0.101)	0.262 (0.093)	**0.811 (0.058)**	0.627 (0.086)
thyroid	−1	0.988 (0.005)	0.988 (0.005)	0.031 (0.005)	0.987 (0.005)
titanic	+1	0.514 (0.079)	0.473 (0.105)	0.556 (0.091)	0.444 (0.127)
titanic	−1	0.676 (0.074)	0.606 (0.090)	0.371 (0.080)	0.711 (0.012)
twonorm	+1	0.915 (0.012)	0.913 (0.018)	0.090 (0.009)	0.917 (0.006)
twonorm	−1	0.913 (0.011)	0.910 (0.015)	0.115 (0.024)	0.911 (0.012)
waveform	+1	0.901 (0.006)	0.899 (0.009)	0.114 (0.005)	0.901 (0.004)
waveform	−1	0.708 (0.025)	0.684 (0.029)	0.248 (0.020)	**0.824 (0.013)**

Table 4.5.: Comparison of the ranks according to AUC in Table 4.4.

Dataset	Class	OMMO	LIBSVM	KPCA	KDE
banana	$+1$	2	3	4	1
banana	-1	3	2	4	1
breast-cancer	$+1$	1	2	3	4
breast-cancer	-1	1	3	4	2
diabetis	$+1$	1	2	3	4
diabetis	-1	2	3	4	1
flare-solar	$+1$	2	3	1	4
flare-solar	-1	2	3	4	1
german	$+1$	2	3	4	1
german	-1	2	3	4	1
heart	$+1$	3	2	4	1
heart	-1	3	2	4	1
image	$+1$	3	2	4	1
image	-1	1	3	4	2
ringnorm	$+1$	3	2	1	4
ringnorm	-1	2	2	4	1
splice	$+1$	2	3	4	1
splice	-1	2	4	1	3
thyroid	$+1$	3	4	1	2
thyroid	-1	1	1	4	3
titanic	$+1$	2	3	1	4
titanic	-1	2	3	4	1
twonorm	$+1$	2	3	4	1
twonorm	-1	1	3	4	2
waveform	$+1$	2	3	4	1
waveform	-1	2	3	4	1
average rank (std.)		2.00 (0.69)	2.69 (0.67)	3.34 (1.19)	1.88 (1.21)

of outlier samples is impossible for most applications, since outlier samples are usually rare.

We use the MIT-CBCL face-detection dataset[3] that contains 2901 images of faces and 28121 images of non-faces of size 19x19 pixels. The dataset is divided into a training set of 2429 faces and 4548 non-faces and a test set of 472 faces and 23573 non-faces. We used the raw data but applied the preprocessing steps described in [94, 45] to reduce the within-class variance; first, we subtracted from each image the gradient of the background to reduce illumination changes by using our novel method described in Section 2.3; second, we performed a histogram equalisation for each image and, third, we scaled each pixel to $[0, 1]$. We applied grid search over the kernel parameter σ and the softness parameter C or ν to obtain the optimal model for the training set. We evaluated the performance for a particular parameter set by choosing randomly 1215 faces to train OMMO and LIBSVM and by evaluating the test error with 1214 faces and 4548 non-faces; to reduce variance we performed 25 runs at all parameter combinations; the performance for a particular parameter set was evaluated by the equal-error rate (EER) of the receiver-operator characteristic (ROC). After we determined optimal parameters, we trained with the whole training set of 2429 faces and computed the ROC curve of the 24045 test samples.

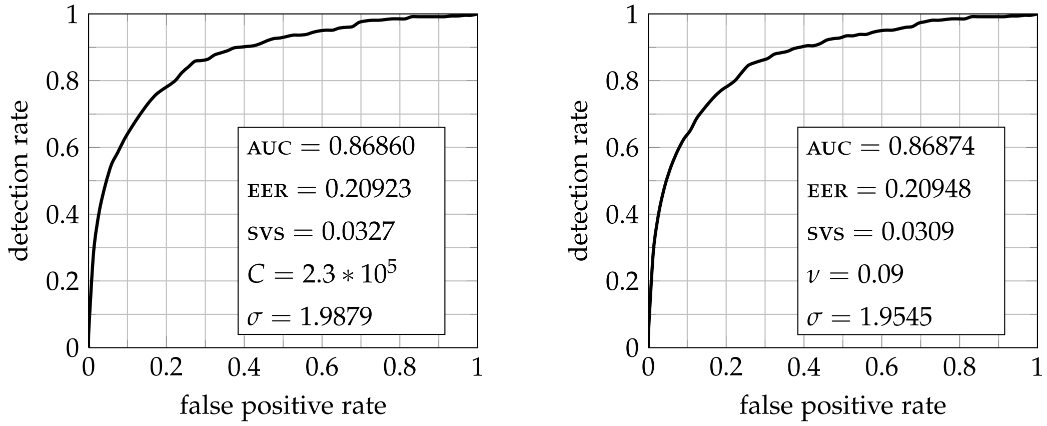
Figure 4.15 compares the ROC curves of OMMO and LIBSVM; even though these two approaches differ significantly in their implementation complexity, they achieve comparable performance for AUC (0.86860 vs 0.86874) and EER (0.20923 vs. 0.20948), which is not surprising since both support-vectors solutions are similar—both yield hard-margin solutions (large C, small ν) and have similar kernel parameters ($\sigma = 1.9879$ vs. $\sigma = 1.9545$).

4.4.6. Fast Model Selection

We have shown that OMMO computes a support-vector solution for novelty detection and performs comparably for various benchmark datasets as well as for the modern application of face detection. In this section, we demonstrate that OMMO with its modifications and a particular strategy qualifies for efficient grid search (model selection).

An optimal grid search should make intensive use of preinitialisation and kernel reuse. So, we want to find a directed spanning tree that has minimum runtime if we travel from the root along the edges. For evaluating the reliability of this method we used 9 benchmark datasets from the same repository we have described previously. Each dataset was separated class-specifically and scaled to unit mean norm and we created a logarithmically scaled parameter grid with 20×20 nodes ($C \in [1, 10^5]$, $\sigma \in [0.1, 5]$) to

[3] http://cbcl.mit.edu/software-datasets/FaceData.html



(a) OMMO

(b) LIBSVM

Figure 4.15.: The ROC curves show that both algorithms achieve comparable AUC and EER performance on a test set of 472 faces and 23573 non-faces; both models obtained by cross validation are hard-margin solutions (large C, small ν) with a similar Gaussian width ($\sigma \approx 1.9$) and almost the same fraction of support vectors.

represent maximum uncertainty throughout the experiment.

A reasonable cost function for describing the computational effort of travelling between solutions of different parameter sets is

$$c(i,j) = \begin{cases} \text{time}_i(j) & \text{if nodes } i \text{ and } j \text{ are neighbours} \\ \infty & \text{otherwise} \end{cases} ,$$

where $\text{time}_i(j)$ describes the runtime of OMMO for grid node i, preinitialised with the results of node j, which itself was trained without preinitialisation. Via the cost matrix, the minimum spanning tree (MST) was determined for every dataset by Edmond's algorithm [31, 95] with each node as a root node. Figure 4.16 shows exemplarily the MST of the banana dataset obtained by using only the positive samples. Most of the edges are oriented vertically rather than horizontally, which indicates that an additional strategy or heuristic for evaluating the parameter grid could reduce computation time considerably.

We evaluate the MST and compare with four heuristics (see Figure 4.17): (i) starting in the top left corner and going to the right and to the bottom (R/B), (ii) starting in the top right corner and going to the left and to the bottom (L/B), (iii) starting in the bottom left corner and going to the right and to the top (R/T), and (iv) starting in the bottom right corner and going to the left and to the top (L/T).

Table 4.6 contains the results of the four heuristics and the MST; apparently, vertical edges in the MST are preferred, i.e. support vector solutions for adjacent kernel widths

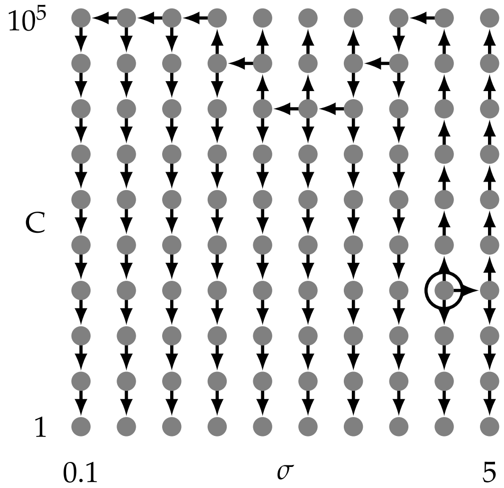
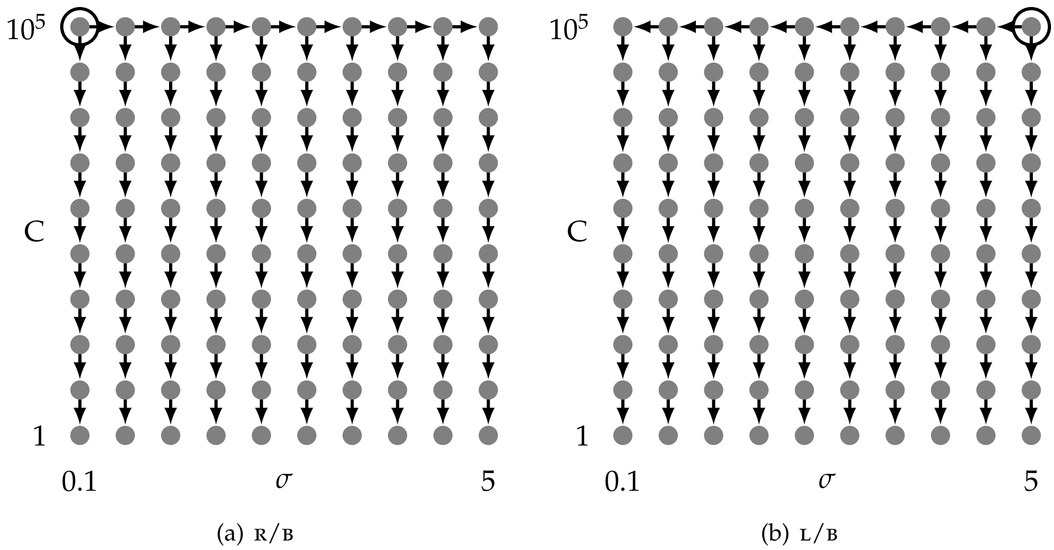


Figure 4.16.: Minimum spanning tree for positive samples of the banana dataset evaluated on the parameter grid described by the kernel parameters C and σ; the root node is depicted by an additional circle.



(a) R/B

(b) L/B

Figure 4.17.: Two of the four heuristics for evaluating the parameter grid; the result of the parent node is used as preinitialisation for every node except the root node; the heuristics R/T and L/T are be obtained by horizontally flipping R/B and L/B.

Table 4.6.: Results of evaluating the parameter grid with different strategies.

Dataset	Class	MST Edges [%]		Computation Time [s]				
		horiz.	vertical	MST	R/T	L/T	R/B	L/B
banana	+1	19	81	69.7	374.3	367.6	93.8	**71.2**
banana	−1	17	83	66.7	366.3	359.0	90.8	**69.8**
breast-cancer	+1	16	84	31.4	215.9	212.2	51.5	**45.3**
breast-cancer	−1	14	86	48.2	348.6	342.9	66.4	**56.3**
diabetis	+1	11	89	60.2	454.2	446.7	87.6	**74.5**
diabetis	−1	9	91	80.4	631.9	622.1	103.6	**85.7**
german	+1	14	86	57.5	470.3	462.1	95.1	**82.0**
german	−1	13	87	85.2	718.7	706.0	133.5	**113.7**
heart	+1	16	84	38.5	287.1	282.1	62.1	**53.9**
heart	−1	14	86	44.5	327.1	321.7	66.4	**57.0**
ringnorm	+1	25	75	147.0	1606.1	1576.3	334.6	**292.2**
ringnorm	−1	17	83	145.3	1617.9	1587.5	230.8	**187.4**
splice	+1	33	67	66.4	796.6	780.8	88.8	**68.1**
splice	−1	28	72	78.8	968.0	948.3	109.1	**83.4**
twonorm	+1	18	82	147.3	1615.5	1585.4	280.1	**237.2**
twonorm	−1	20	80	164.3	1836.6	1771.4	310.2	**201.1**
waveform	+1	18	82	97.2	1039.8	1020.2	175.4	**147.0**
waveform	−1	17	83	138.9	1492.7	1464.3	246.6	**206.5**
median		17	83	74.3	675.3	664.0	99.4	**83.4**

differ more than those of adjacent softness parameters. Moreover, the heuristic L/B (see Figure 4.17(b)), which starts in the top right corner and goes to the left and to the bottom, significantly outperforms the other two strategies. Moreover, the results indicate that strategy L/B is very close to the optimal strategy—the MST.

The probabilities of the four orientations of edges in the MST differ significantly, see Table 4.7. Since most of edges are oriented towards the bottom and the left of the grid, the heuristic L/B is reasonable. We obtain the best improvement when training along the MST, but according to Table 4.8 the mean relative computation time of L/B is 5.9% compared to the time without grid strategy, which corresponds to an improvement by a factor of approximately 17 and which is close to optimum (5.3% for the MST).

Finally, we can observe that with optimisations such as the feasibility gap as stopping criterion, preinitialisation, and a particular grid search strategy, OMMO qualifies for extremely fast model selection such that the computation for a complete grid search is reduced to only 5.9% of the full search, which is close to the optimal solution.

Table 4.7: Mean probability of edge orientations in the MST over all datasets. Edges that are oriented towards the bottom of the grid, i.e. decreasing C and constant σ, appear most often in the MST, whereas edges to the right, i.e. increasing σ and constant C, are rarely used in the MST.

Edge Orientation	Mean Probability [%]
Left	11.2
Right	5.8
Top	20.6
Bottom	62.4

Table 4.8: Computation time of the four heuristics and the MST scaled to the mean runtime without applying a particular strategy. All heuristics improve the computation time for grid search; the heuristics going from bottom to top, i.e. R/T and L/T, reduce the computation time by a factor of approximately 2, whereas the heuristic in the opposite direction L/B yields an improvement by a factor of almost 17.

Strategy	Mean Relative Runtime [%]
None	100.0
R/T	48.2
L/T	47.3
R/B	7.0
L/B	5.9
MST	5.3

4.5. Discussion and Outlook

We have introduced a novel, efficient support-vector approach for novelty detection called OneClassMaxMinOver (OMMO). The OMMO algorithm employs a perception-like learning rule that performs a learning step for samples farthest from the current decision boundary and a forgetting step for samples closest to the current decision boundary. In its simplest variant, OMMO requires only a few lines of code; in the case of MATLAB, for instance, it takes less than 10 lines of code with only simple operations such as dot-products. Hence, OMMO can even be used and integrated without specific knowledge in optimisation theory.

Theoretically, we have shown that OMMO converges to the maximum margin solution that is solely based on support vectors. Moreover, we have performed extensive experiments on various benchmarks datasets, where we compared OMMO with LIBSVM, a modern toolbox for obtaining support vector solutions, a kernel density estimator (KDE) and an approach based on kernel principal component analysis (KPCA). The results have demonstrated that OMMO yields comparable performance regarding computation time and classification error compared to LIBSVM in most cases; for few datasets, especially in high dimension, OMMO significantly outperforms LIBSVM. The OMMO algorithm achieves accurate results over almost all benchmark datasets, whereas the performance of KPCA and KDE strongly depends on the characteristics of the dataset.

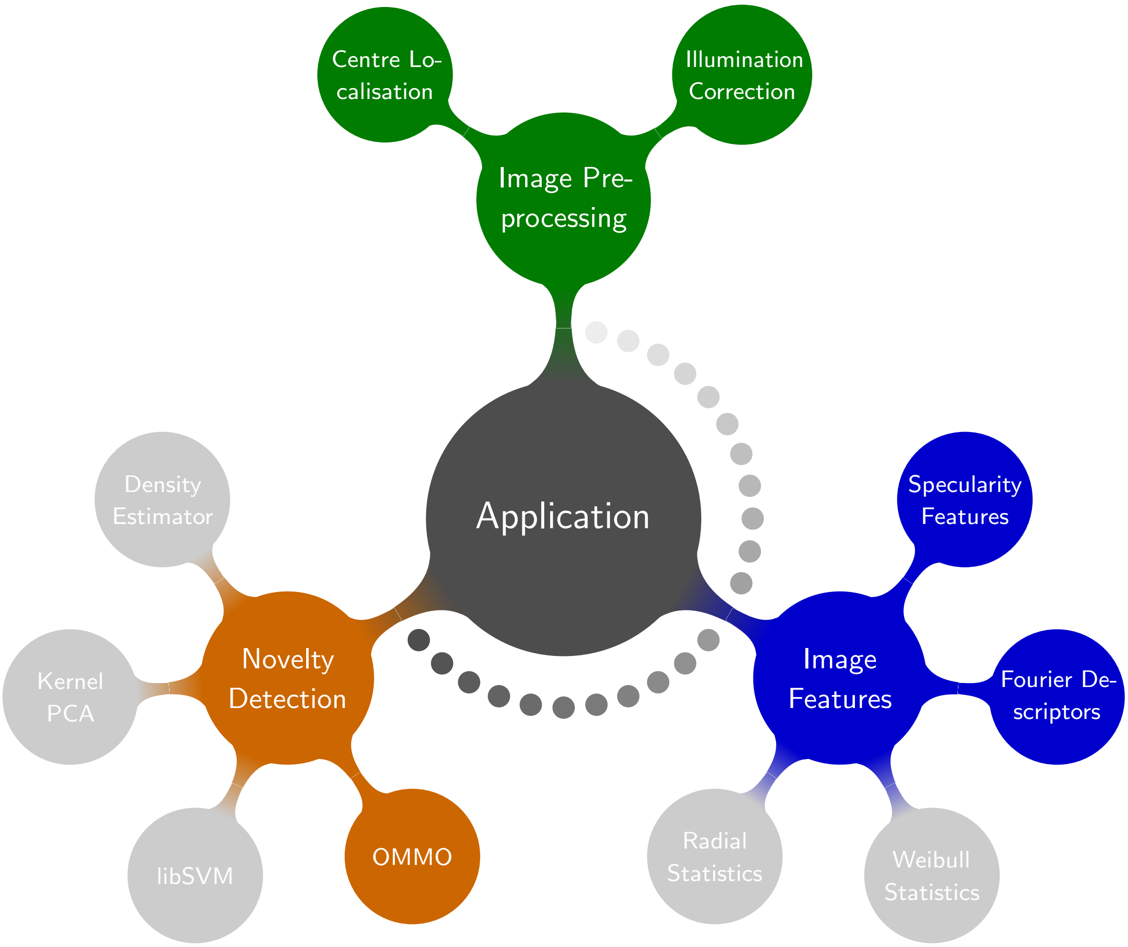
Furthermore, we have introduced several optimisations for OMMO such as kernel caching, cache reuse, alternative stopping criterion or preinitialisation, which qualifies OMMO for large-scale high-dimension problems. We have demonstrated that these optimisations, in combination with a particular grid search strategy, reduce the computation time of OMMO by a factor of 17; hence, model selection for OMMO can be performed efficiently.

There are still some modifications that can be applied to the OMMO algorithm. For example, outlier samples could be integrated already into the training as it is done by the SVDD approach [96, 97, 114]. However, experiments need to be performed to verify whether a similar behaviour could be obtained by adapting the threshold of the hyperplane such that these outliers are outside the data description. Recently, Liu et al. [71] have proposed a modification to SVDD that considerably reduces the time complexity for the classification of new samples after the decision boundary has been learnt. Since the kernel expression in the decision function contains a linear combination of support vectors, the runtime complexity grows linearly in the number of support vectors. This kernel expansion, however, can be replaced by a single feature vector that is obtained by pre-image techniques and by using simple relationships between this vector and the centre of the hypersphere; the decision function is, then, no longer linear in the number of support vectors, but is constant. It is a question for future research to investigate whether this optimisation can also be applied to the OMMO algorithm.

Part II.

Machine Vision Applications

5. Welding Seam Inspection



This chapter demonstrates the application of specularity features, statistical Fourier descriptors, and OMMO to the problem of welding seam inspection. The data has been provided by Philips GmbH, GTD Mechanisation, Aachen. Some of the work described in this chapter has been previously published in [103, 105].

5.1. Introduction

In many industrial processes, such as printed circuit board assembly or automotive line spot welding, individual parts are joined by soldering or welding techniques. The quality of a single welding often defines the grade of the whole product; in areas such as the automotive or aviation industry, for example, critical failures of the welding process can cause a malfunction of the whole product. Typically, welds are made by a soldering iron or a laser, which has recently become more affordable. Even though the initial cost of a laser-welding system is still high, their wear-out is low and the service intervals are long. A laser weld is more precise than a weld by a soldering iron, but the quality can also vary due to shifts of the part towards the laser or due to material impurities. An inspection of the welding is hence required to guarantee high quality.

Several machine-vision approaches for automatic inspection of solder joints have been proposed; they can be divided into two groups. Approaches of the first group focus on the development of specific camera and lighting settings to gain the best image representation of relevant features [79, 54, 21]; approaches of the second group must use a fixed camera and lighting setting and comprise sophisticated pattern recognition methods [58, 82, 79, 28, 54].

Here, we describe a machine vision system for the inspection of cathodes welded by an Nd:YAG (neodymium-doped yttrium aluminium garnet) laser during the production of Xenon lamps. We, first, extract the regions of welding seams, for which we, then, compute characteristic image features that capture the specular reflections of defective welding seams; finally, we perform novelty detection based on these image features.

5.2. Image Acquisition

An unwelded cathode consists of a socket and a pole that may be composed of different materials, see Figure 5.1. In a top view with directional parallel light, the unwelded cathode simplifies to only four components—two black rings (the slant of the neck and the space between pin and socket), one white ring (neck of the socket), and one white circle (top of the pin). The analysis of connected components of the welded cathode can therefore be used to extract specific features.

A correct combination of camera, lens, and illumination is very important to achieve the best performance in classification. However, sometimes the best setup cannot be chosen due to limited space or other requirements. We were limited to only one camera setup for this work (see Figure 5.2) and we used a standard analog monochrome VGA video camera, a single-sided telecentric lens, and an LED ring light with a Fresnel lens. The images we captured with this setting contain the cathode as well as its surrounding area; we, therefore, applied our novel centre localisation approach, proposed in Section 2.2.2, to extract only the cathode region.

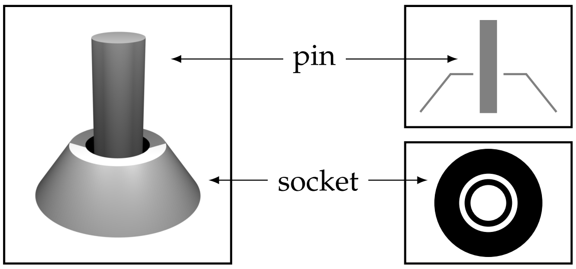


Figure 5.1.: Left: 3D drawing of the cathode. Upper right: cross-section. Lower right: top-view.

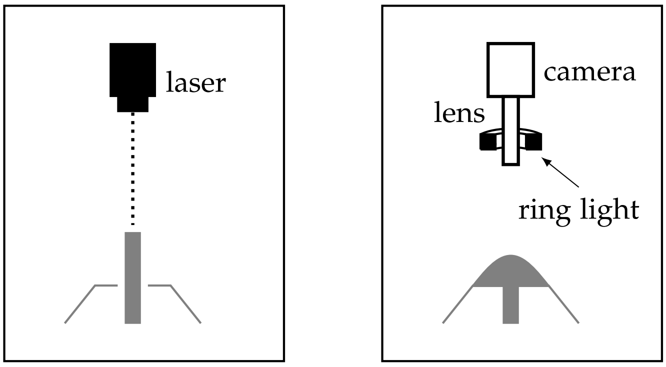


Figure 5.2.: Drawing of the setup for laser welding (left) and image acquisition (right). The laser and the camera are located on top of the cathode. A Fresnel lens was mounted in front of the ring light to obtain bright illumination at the cathode.

Moreover, we removed images of unwelded cathodes and images that are classified easily to reduce the amount of images. The dataset, thus, consists of only images that are difficult to classify manually. In total, we collected 934 images containing 657 images of defect-free cathodes and 277 images of defective cathodes. Each image was labelled by experts, scaled to different sizes (10×10, 20×20, 40×40, 80×80 px), and smoothed by a Gaussian filter ($\sigma = 1$) to reduce noise. Since we apply feature-extraction algorithms that use raw pixel intensities and that are not invariant towards rotation, we rotated each image four times ($0, \frac{1}{2}\pi, \pi, \frac{3}{2}\pi$) giving a total amount of 3736 images.

The true class labels are generally unknown, since defective cathodes are determined by the mean time to failure, which cannot be measured during manufacturing. Experts, thus, look for arbitrary deviations from typical defect-free samples that have been selected by extensive benchmark tests; Figure 5.3 depicts some of these examples. Defect-free samples show various reflections, which result from material impurities or from an imprecise position of the pin. Moreover, a slight deflection of the pin just before the welding affects not only the appearance but can also yield defective cathodes. Some of the defective cathodes have holes caused by a slanted pin, others do not have any reflections due to a very rough surface. We, therefore, cannot describe the variety of

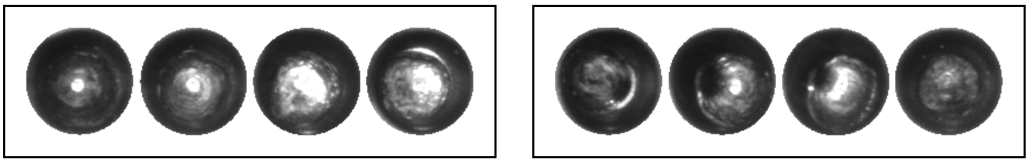


Figure 5.3.: Example images of defect-free (left) and defective cathodes (right); the region-of-interest was extracted with our novel gradient-based approach for centre localisation proposed in Section 2.2.2. Note that the differences between the two classes is not obvious.

defects completely, and a novelty-detection system must be applied. Moreover, features that cover characteristic specular reflections are required.

5.3. Methods

Recently, several approaches for the inspection of solder joints have been proposed [79, 21, 58, 54, 28]. Some of these methods compute simple features in a manually tiled binary image, others use the pixel intensities directly as input features for a neural network or a support vector machine [24, 12, 110]. Since the number of pixels is very large compared to the number of data samples, preprocessing often involves a downsampling of the images to reduce the dimensionality considerably; this downsampling, however, reduces the information contained in the images and therefore yields poor error rates; in contrast, specifically arranged and compressed pixel intensities yield superior performance. We will demonstrate that specifically designed features that capture properties of specular reflections perform best.

5.3.1. Feature Extraction

We apply the two feature sets that have been presented in Section 3.2—specularity features (SPEC) and statistical Fourier descriptors (SFD). The SPEC features describe statistics of specifically designed shape characteristics based on a stack of binary images and can cover a wide range of complex shape properties and their dependencies. In contrast, the SFD feature set captures shape statistics that have not been specifically designed for the application, but it covers a wide range of characteristic shape properties using the Fourier transform of the component's boundary. We compare with raw pixel intensities (RAW), radial encoded raw pixel intensities (RADIAL, proposed in Section 3.4), and the statistical geometric feature (SGF) algorithm that computes simple geometric properties of binary components and that has successfully been applied to the problem of tissue classification [112].

For the SFD, we apply equal arc-length sampling of the boundary with 128 sampling points and compute the phase and the magnitude of the first 64 Fourier coefficients as local shape features. Since position, rotation, and size of components may be relevant properties, we only scale the Fourier coefficients to be invariant towards the starting point of the shape signature, see Equation 3.9.

Moreover, we use different image depths ranging from 2 bit to 8 bit, since the performance of SPEC, SFD, and SGF can vary depending on grey level quantisation.

5.3.2. Novelty Detection

Standard two-class classifiers require samples that describe both classes in a proper way. In our case, however, there are only few defective cathodes that are characterised well and the classes are imbalanced (2/3 vs. 1/3). We, therefore, apply our simple and incremental training algorithm for support vector data description with several improvements for fast parameter validation—see Algorithm 4.2, [64, 102]. In contrast to standard two-class classifiers, which separate the input space into two half-spaces, our novelty-detection approach learns a subspace such as to enclose the samples of only the target class as tightly as possible; this not only increases robustness against unknown types of outliers significantly, but also extends the time intervals for retraining when new samples are available.

We choose a Gaussian kernel and perform model-selection using 10-fold cross validation. We further scale the input features to zero mean, unit variance, and unit mean norm. Moreover, we perform simple feature selection by removing every integer-valued feature that takes less than two values to speed up the novelty-detection algorithm. Finally, we apply a Wilcoxon signed rank test to the test errors to compare the different feature extraction methods.

5.3.3. Feature Analysis

Most of the techniques we use for feature extraction generate high-dimensional data, especially if we use the raw pixel intensities. By analysing these feature vectors we want to address two aspects; first, we want to detect features that contain almost no information and remove those features to save memory and computation time; second, we want to evaluate those raw features, SGFS, and SPECS that are most discriminative for the description of defect-free and defective weldings. We, thus, can verify whether the specifically designed features can accurately describe the characteristic specular reflections and the physical shape of the cathodes. We use two different approaches for evaluating the feature set—principal component analysis (PCA) and linear discriminant analysis (LDA). In case of PCA we take only defect-free samples and evaluate those principal components that show the largest absolute eigenvalues. Even though the class of defective weldings is not sampled properly, we apply LDA and sort the entries of the resulting weight vector according to their absolute value.

5.4. Results

5.4.1. Results of Feature Analysis

We analysed the raw pixel intensities using PCA as described above; the resulting Eigenimages show three important aspects (see Figure 5.4). First, pixels in the centre are more important than pixels at the border of the pole; this corresponds to the description of the cathode, where white reflections (regions) in the centre of the image indicate a defect-free welding. Second, the ring structure, i.e. the area at the neck of the socket, also shows high relevance. Third, more complex geometric shapes are significant (see Figure 5.4, second row). Based on the observation of the Eigenimages, we can expect that radially encoded pixel intensities are more appropriate and will yield better performance than raw pixel intensities without specific organisation. However, it remains unclear whether a sophisticated feature extraction approach can capture complex geometric properties more accurately than any specific organisation of pixel intensities.

We analysed the discrimination performance using LDA, which computes a linear hyperplane such that the classification error that is obtained when projecting both classes onto this hyperplane is minimised; we ranked the features according to their absolute value in the normal vector of the LDA hyperplane. Some of the most significant SGFS are: (i) the sample standard deviation of irregularity of white components, (ii) the mean of irregularity of white components, (iii) the sample standard deviation of total clump area of white components, and (iv) the mean of displacements of white

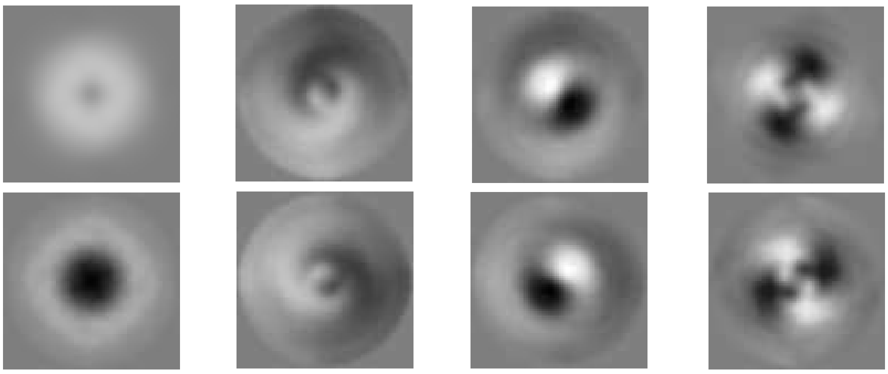


Figure 5.4.: Eigenimages with large Eigenvalues that result from images of welded cathodes of size 80×80 px and 8 bit grey level depth. Relevant pixels are depicted by large and small values (bright and dark), whereas irrelevant pixels are depicted by mid grey, i.e. 128. Obviously, relevant pixels are not only in the centre or in a ring around the centre (first column), but also pixels in opposite regions are relevant (third and fourth column); this indicates that a specific organisation of raw pixel intensities, such as radially-encoded pixel intensities, will describe the image characteristic more appropriate than raw pixel intensities without specific organisation.

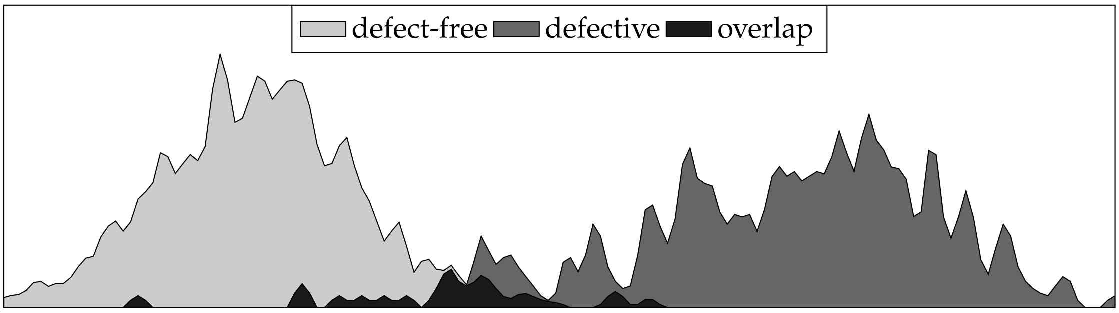
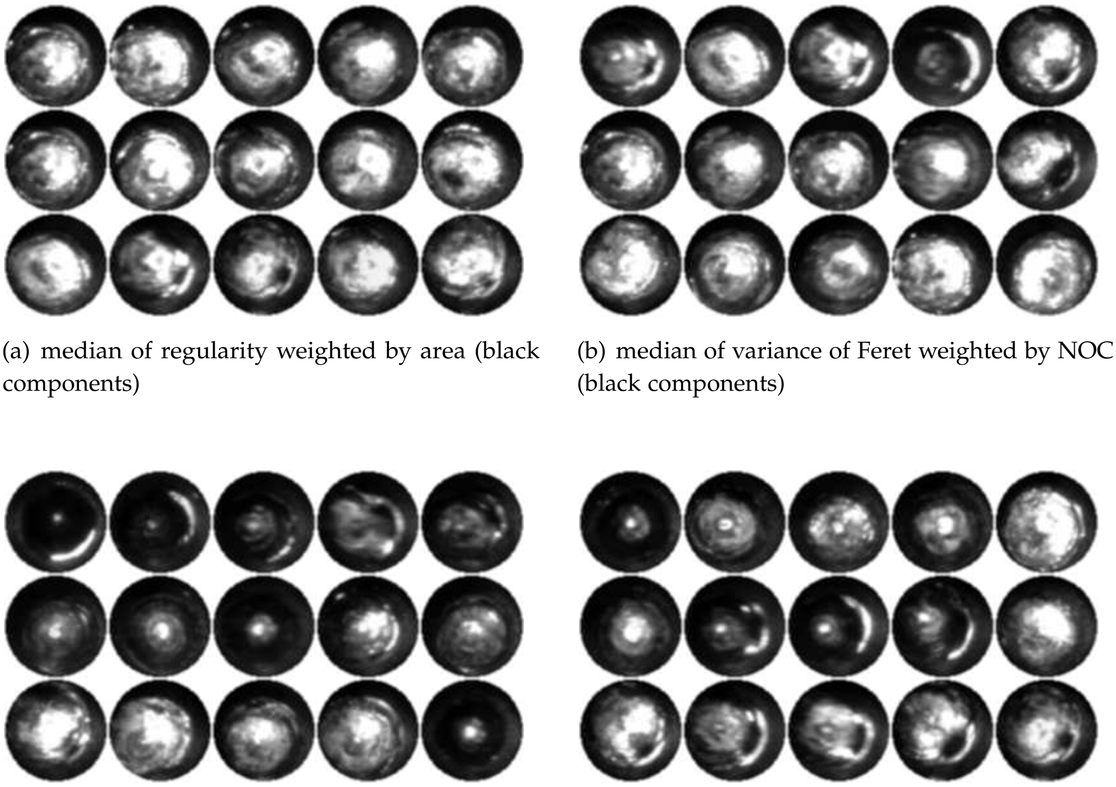


Figure 5.5.: Class distributions when projecting the SGF features (80×80 px, 8 bit) onto the hyperplane obtained by LDA.

components. This shows that features of white components are more important than those of black components. Furthermore, the shape of the components, their position, and their size are relevant properties for the discrimination. If we project the SGFs of both classes onto the hyperplane obtained by LDA both classes overlap (see Figure 5.5); thus, in the feature space of SGFs there exists no hyperplane that can separate both classes perfectly.

Some of the most important SPEC features when using LDA are: (i) the median of regularity of black components weighted by their area (see Figure 5.6(a)), (ii) the median of variance of the Feret diameter of black components weighted by the number of components (see Figure 5.6(b)), (iii) the variance of distances from the centre of the white components weighted by the number of components (see Figure 5.6(c)), and (iv) the sample standard deviation of the area of the bounding rectangle of black components weighted by the number of black components (see Figure 5.6(d)). This indicates that black components become important when using local features such as the Feret diameter or the bounding rectangle. Since the SGFs are limited to only a few local features of white components, properties of black components are completely ignored. Moreover, the area of a component is also relevant as it is used for the scaling of the local features. This becomes more apparent by analysing the ranking of the 100 most relevant SPECs; there, 45 local features are scaled by their area and 55 are scaled by the number of components. Moreover, 43 local features correspond to black components and 57 correspond to white components; this demonstrates that properties of the specular reflections cannot be described accurately by only using a particular component colour or a single scaling method. If we project SPEC features onto the LDA direction, the overlap becomes smaller compared to the SGF, which indicates that the discrimination power of SPEC is significantly higher (see Figure 5.7). However, there is also no hyperplane that separates both classes perfectly.

In contrast to SGF and SPEC, we cannot interpret the SFD features and make a comparison with the specular reflections of the welded cathode. Since we employ a Fourier



(a) median of regularity weighted by area (black components)

(b) median of variance of Feret weighted by NOC (black components)

(c) variance of distances from centre weighted by NOC (white components)

(d) sample standard deviation of area of bounding rectangle weighted by NOC (black components)

Figure 5.6.: Example images with large values of the indicated features (top row), medium values (middle row) and small values (bottom row).

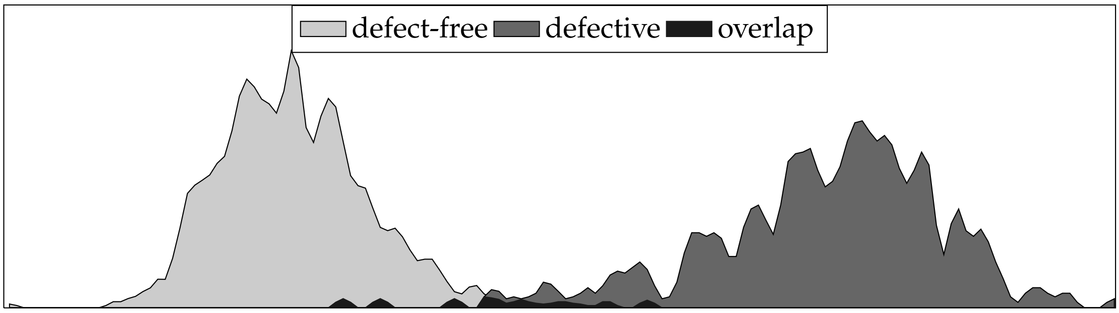


Figure 5.7.: Class distributions when projecting the SPEC features (80×80 pixel, 8 bit) onto the hyperplane obtained by LDA.

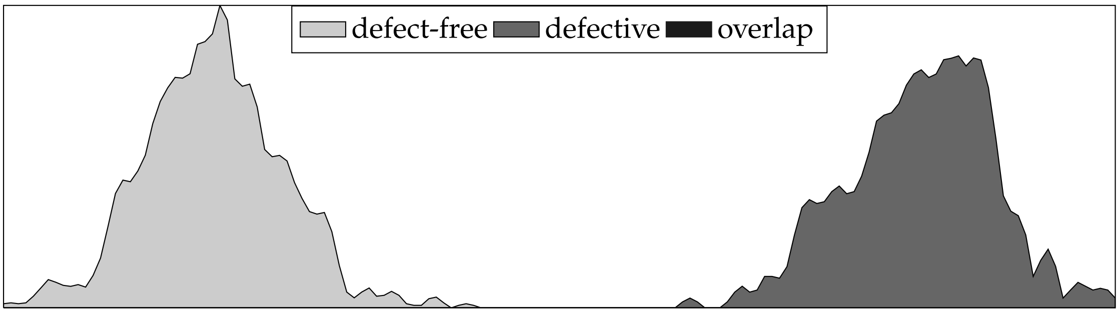


Figure 5.8.: Class distributions when projecting the SFD features (80×80 pixel, 8 bit) onto the hyperplane obtained by LDA.

transform of the object's boundary we can no longer identify a single feature that, for example, captures the compactness of the object—instead, the SFD approach computes statistics of all possible boundary shapes by analysing and combining the coefficients of the Fourier transform. Figure 5.8 shows the LDA analysis for the SFD features, where we observe that both classes can be perfectly separated by a hyperplane. This demonstrates the superior performance of the SFD features compared to SGF and SPEC and it shows that, in case of welded cathodes, the specular reflections and their dependencies can be described more accurately when general boundary characteristics are computed instead of specifically designed characteristics.

5.4.2. Results of Classification

Table 5.1 shows the classification results for the different feature extraction approaches; we can observe four major aspects. First, radially encoded raw pixel intensities perform significantly better than raw pixels without any organisation; this neither depends on the image size nor on the number of grey levels. The best performance of radially encoded raw pixels is obtained using images of size 40×40 pixels and 16 grey levels (error of 13.5%).

Second, SGF as well as SPEC improve the classification performance considerably, compared to RAW and RADIAL features. The best performance (5.3% error) of the SGF feature set is obtained with images of size 80×80 pixels and 256 grey levels; for the same image size and the same number of grey levels SPEC significantly outperforms SGF (3.8% vs. 5.3% and with $p = 0.03$). We can further observe that the superior performance of SPEC compared to SGF also holds for smaller images (40×40 pixel), which demonstrates that the SPEC features can describe properties of specular reflections more properly than SGF features or radially encoded raw pixels. It further shows that a large number of grey levels is required to describe specular reflections in both cases, for SGF as well as for SPEC.

Third, SFD features yield the lowest error rate by far (0.9 %) using large images

Table 5.1.: Classification performance of the different features sets in welding seam inspection. We computed the median error rate (in %) over 10 test sets and the minimum median error rate for each feature set is highlighted. We omit results for smaller images, e.g. 20×20 pixel, in case of SGF, SPEC, and SFD, since they do not yield any improvement.

feature set	number of grey levels						
	4	8	16	32	64	128	256
RAW (20×20 pixel)	20.3	17.6	17.5	18.0	18.0	18.4	**17.5**
RAW (40×40 pixel)	22.4	18.9	**17.2**	18.0	17.5	17.5	17.6
RADIAL (20×20 pixel)	14.8	15.7	14.9	15.3	**14.4**	14.8	14.8
RADIAL (40×40 pixel)	14.9	14.4	**13.5**	14.9	15.3	15.4	15.4
SGF (40×40 pixel)	15.7	10.8	8.9	7.1	**6.3**	6.8	6.8
SGF (80×80 pixel)	7.6	7.6	6.8	6.7	6.2	5.8	**5.3**
SPEC (40×40 pixel)	11.9	8.1	5.8	5.5	**5.0**	5.1	5.2
SPEC (80×80 pixel)	9.1	6.9	5.8	5.6	5.3	4.0	**3.8**
SFD (40×40)	6.2	6.0	5.4	5.5	4.2	3.8	**3.5**
SFD (80×80)	5.1	5.1	4.5	3.2	2.1	1.4	**0.9**

and maximum number of grey levels; even for small images with few grey levels SFD yields superior performance compared to the other feature sets. This demonstrates that the complex specular reflections of welding seams can be captured accurately by a Fourier analysis of the component's shapes. Note that we do not obtain zero error rates, since we have optimised for a high detection rate of defective samples by computing a compact data description of defect-free samples—in contrast, the LDA analysis was performed using the complete dataset. However, we expect that the estimated error will decrease as the number of available data samples increases.

Fourth, we cannot observe any significant difference for different image sizes when using raw pixel intensities—in contrast, the error rate decreases as the image size increases for SGF, SPEC, and SFD. However, we can identify that grey level resolution is more important than spatial resolution. Since the structures of the welded pin and the socket are merged such that holes and rings cannot be detected any more, the relationship between grey level depth image size only holds for larger images, e.g. larger than 20×20 pixels.

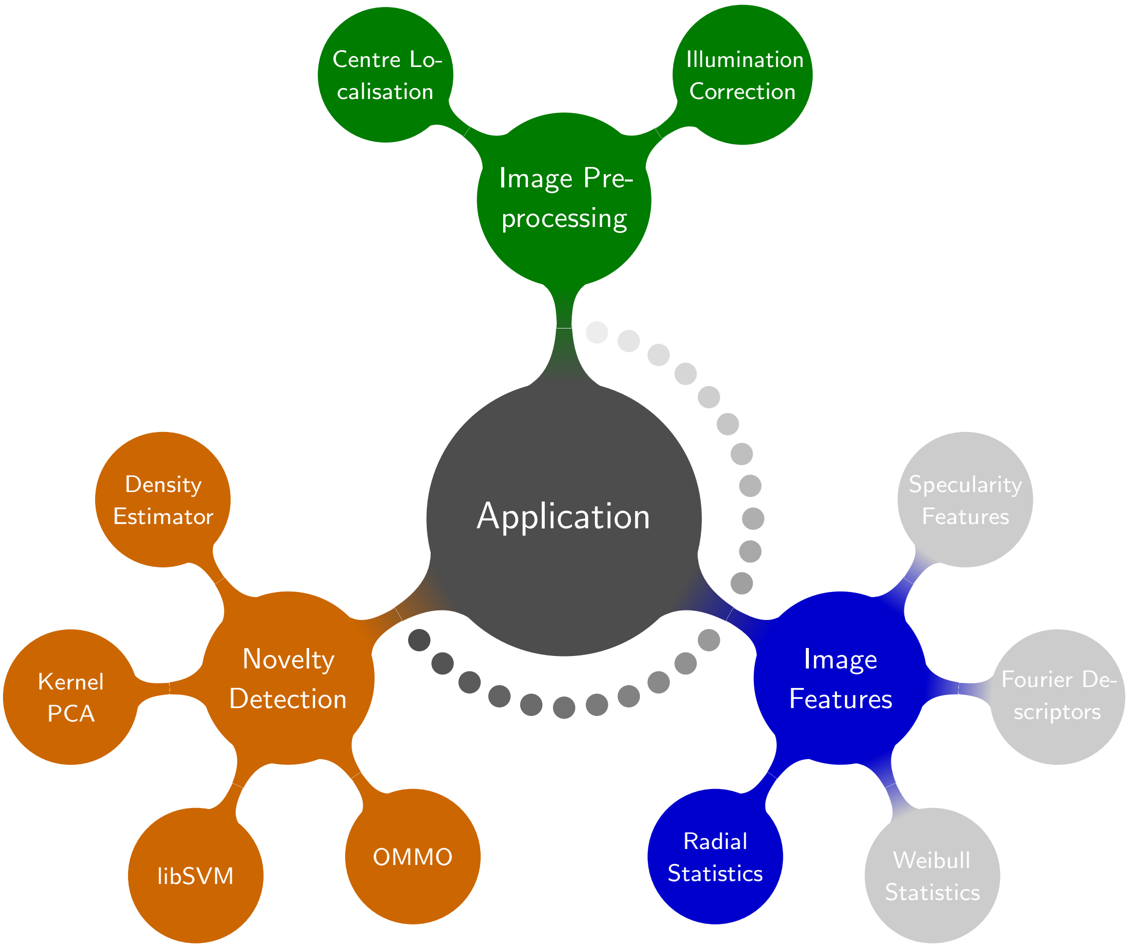
5.5. Discussion

We have shown that welding seams can successfully be inspected with our novel feature set, called statistical Fourier descriptors (SFD). These features significantly outperform raw pixel intensities with and without encoding and more sophisticated feature sets such as statistical geometric features SGF or specularity features (SPEC). We determined relevant locations in the image by analysing raw pixel intensities and relevant features of SGF and SPEC, and we found white regions in the centre of the image and their shape to be of high importance for the classification. The SPEC feature set can cover several complex shape properties and their dependencies and is, nevertheless, intuitive and computed efficiently. In contrast, when computing SFD we can no longer interpret the features and the time complexity increases due to the Fourier transform, but we achieve significantly better performance. Both SPEC and SFD are well appropriate for the automatic inspection of welding seams and can even be applied to a wider range of machine vision problems concerning complex specular reflections, such as surface inspection or defect detection with specular objects.

In case of SFD, it is a question for future research to investigate whether the number of sampling points and the number of used Fourier descriptors strongly influence classification performance. Moreover, we believe that SFD could also be applied to other applications such as to the problem of texture classification or to the problem of image content classification to organise image databases.

The labelling of industrial datasets such as images of solder joints or other weldings is usually based on experts viewing images and not on the actual functional test. Hence, these labels are very subjective and do not necessarily correspond to the physical and electrical properties of the weldings. Therefore, additional information about the welding such as conductivity, rigidity, or weld strength has to be collected and combined with a machine-vision based approach to yield further improvement. However, the error rate of SPEC and SFD features are already comparable to those obtained by manual inspection, as reported by internal studies.

6. Inspection of Light-Emitting Diodes



This chapter demonstrates the application of radial statistics and the novelty-detection methods described in Part I to the problem of defect detection in images of LEDs. The data has been provided by Philips GmbH, GTD Mechanisation, Aachen. Some of the work described in this chapter is part of [98].

6.1. Introduction

Over the last few years, light emitting diodes (LEDS) have been employed in various industrial products and have recently become popular due to the increased demand for energy-saving television sets. Similar to Moore's Law for transistor integration in integrated circuits, there is the so-called Haitz's Law [41] for LED devices, which states that every decade the light output level of a LED device increases by a factor of 20 and that the cost per lumen falls by a factor of 10. It is therefore expected that LEDS will soon be the dominant technology for many lighting applications.

Like in many other production processes an inspection of LED is required for quality assurance; this inspection must be performed either electrically or visually. Electrical inspection ensures correct functionality, but since an extensive stress test can only be applied to few LEDS, defects that might cause a malfunction after a period of time cannot be detected with high reliability. Therefore, a visual inspection must be used to detect potential short-term, long-term, and cosmetic defects. However, manual visual inspection involves significant labor and production costs. Even though human experts are very flexible to variations of the production process, different experts may obtain different results (inter-observer variability) or even one expert may obtain different results for the same sample (intra-observer variability). Furthermore, visual inspection may lead to misjudgements due to human fatigue [113]. These shortcomings require an automatic visual inspection for LED manufacturing.

In this chapter, we focus on the inspection of high-power LEDS, which are used, for instance, in automobile head- and backlights. The die of a high-power LED mainly consists of three components: light emitting area, disjunctions, and p-electrodes—as shown in Figure 6.1. Besides several cosmetic defects, one of the most important defects concerns the p-electrode and its surrounding area. If one of the 16 disjunctions is slightly damaged or broken, the p-electrode gets directly connected to the light emitting area and the LED will not function. Furthermore, there is a smooth transition between immediate malfunction and future malfunction of the LED and it is directly correlated to the damage of the disjunction. Thus, this defect type has to be detected as early as possible. Since the damage of the disjunction is a critical defect, an automated machine vision system must satisfy two major requirements: (i) 100% detection rate for defects (true negative rate) and (ii) minimum false alarm rate (false negative rate). Since manufacturing of high-performance LED produces approximately 1 LED per second on each machine, a false alarm rate of 1% may yield over 300.000 rejected LED per year. Manufacturing of a particular product is often performed on several machines simultaneously to increase production; then, the number of defect-free LED wrongly classified as defective may rapidly exceed a million per year.

Figure 6.2 shows example images of defective LED; due to strong image noise and blurring, some defects are even difficult to detect manually. Simplified, a defect occurs

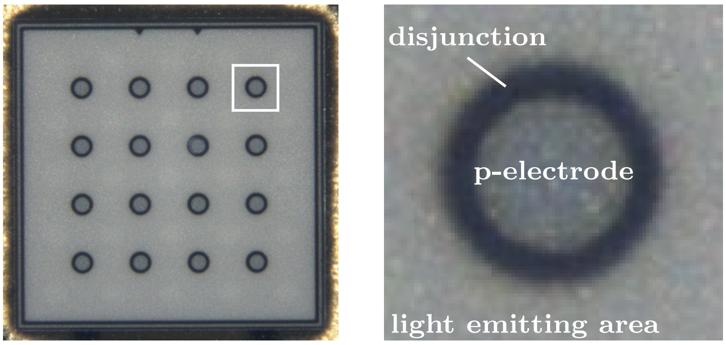


Figure 6.1.: Left: Example image (1024 × 1024 px) of an LED die with a size of 1 mm^2. Right: Zoomed area around a p-electrode. The die mainly consists of three components: light emitting area, disjunctions, and p-electrodes. The disjunction, which includes a non-visible dielectric, has a width of $8\,\mu$m and an outer radius of $30\,\mu$m. The image resolution is approximately $1\,\mu$m/px.

if the dark ring (disjunction) is interrupted. Therefore, we evaluate several image features and classification approaches to obtain an inspection system that can detect such discontinuities with high accuracy while preserving simplicity and efficiency.

Recently, some approaches for automatic inspection of LED or similar inspection tasks such as inspection of semi-conductor components have been proposed; Chen and Hsu, for instance, have employed simple current-voltage statistics as features for LED inspection [19]; others perform blob analysis on a single binary image or a series of binary images [105, 33, 17, 16]. Within the group of spectral approaches, wavelet characteristics have been used as features [69, 70]; a good survey on spectral approaches for texture classification, which is similar to defect detection since a defect somehow changes the underlying texture, can be found in [83], for example. In the case of model-based approaches, the comparison with a defect-free reference image, often called *golden template*, is one of the simplest approaches [40, 121, 22]; comparison can be performed by cross-correlation or statistical analysis, for instance. Even though reference-based methods are simple and intuitive, there are several issues that strongly influence performance in real applications. For example, in the case of highly structured surfaces an expensive registration must be performed beforehand to compare with the template. Furthermore, strong noise in combination with complex texture can affect the comparison such that subtle defects are missed.

Many of these approaches were developed and optimised for a particular problem at hand and their performance on novel datasets is very limited. Randen and Husoy [83] evaluated different methods for texture classification and found that their performance strongly depend on the image type. Hence, they did not identify a single approach that performs best on all datasets, which suggests that a powerful feature-extraction method always incorporates prior knowledge, especially in the case of defect detection with subtle and arbitrary defects.

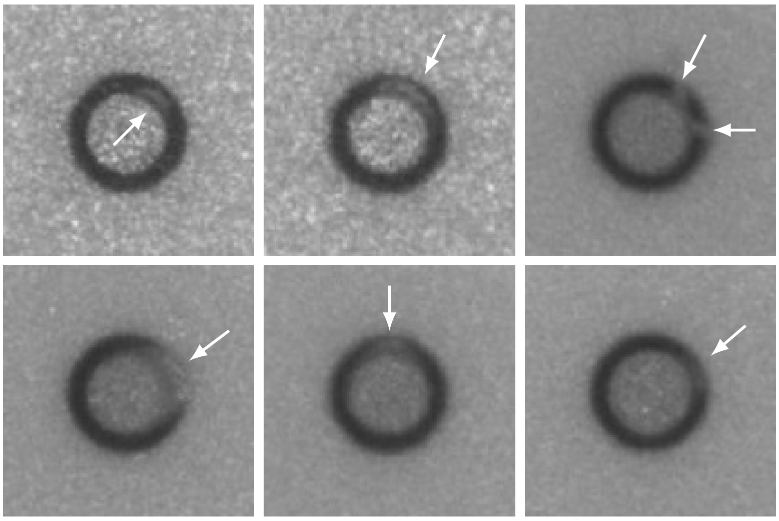


Figure 6.2.: Example images of LED with defects (white arrows) at the disjunction. Some of the defects are clearly visible, for others human experts have to look carefully. Some images contain strong noise, others are blurred due to a wrong focus. The poor image quality makes manual inspection even difficult. Note that some defects may be hardly visible on printed paper due to low printer resolution.

6.2. Methods

6.2.1. Preprocessing

Each die contains 16 p-electrodes placed in a 4-by-4 grid, which we must extract with high accuracy to apply our radial encoding schemes proposed in Section 3.4. Since the position of this 4-by-4 grid varies only slightly, we first estimate rough position of the p-electrodes (see Figure 6.3). Then, we localise the correct centre of each p-electrode by applying our novel symmetry-based approach for accurate centre detection, which has been described in Section 2.2. Compared to standard techniques for centre detection such as the Hough transform our approaches achieve superior performance regarding accuracy, robustness, and efficiency (see Table 2.1). Based on the estimated centre we extract a region of 100×100 pixels to cover the p-electrode and its surroundings. Moreover, we apply our method for illumination correction, see Section 2.3, to remove inhomogeneities.

6.2.2. Feature extraction

We apply our two novel radially-encoded feature sets, proposed in Section 3.4. The first set consists of radial segments, for which mean and standard deviation are evaluated; the second is based on sampled orientations. For both we combine the seg-

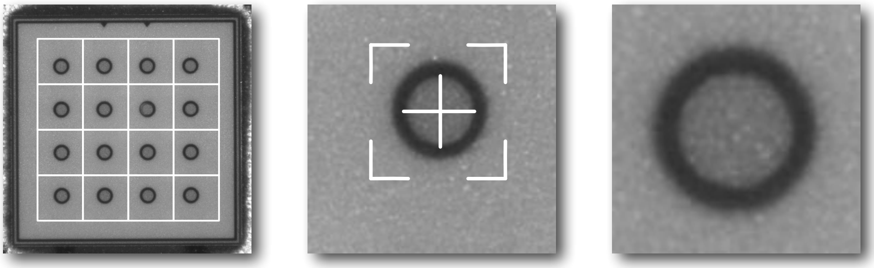


Figure 6.3.: The preprocessing steps from left to right: (i) rough estimation of the centres by placing a virtual grid on the die and extraction of the grid elements, (ii) high performance centre localisation, and (iii) extraction of a 100×100 pixel region of the p-electrode.

ment/orientation characteristics by computing first order statistics such as maximum and mean. Since defects of the p-electrode affect the appearance of the dark ring such that it will be distorted or even broken (see Figure 6.2), statistics of raw pixel intensities that have been radially organised are expected to yield superior performance compared to pixel intensities without radial organisation.

Pearson's correlation coefficient

The comparison of an input image with a reference image is very intuitive, since this is similar to what human experts perform; based on a number of defect-free samples human experts recognise a defect as a deviation from already known samples.

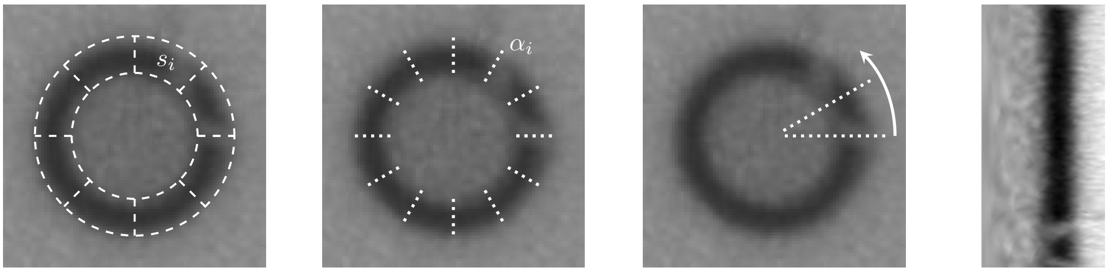
Pearson's correlation coefficient is often used for template/reference matching due to its robustness to brightness variations and noise. If template T and the image I have the same width W and height H, then Pearson's correlation coefficient c is defined as:

$$c(I,T) = \frac{1}{\sigma_I \, \sigma_T} \sum_{i=1}^{H} \sum_{j=1}^{W} (I(i,j) - \mu_I)\,(T(i,j) - \mu_T) \quad , \tag{6.1}$$

where μ_I and μ_T are the image mean and the template mean, and σ_I, σ_T are the standard deviations of the image and the template. If template and image are identical, the correlation coefficient takes its maximum value ($c = 1$); for perfect anti-correlation, which means that the template is identical to the inverted image, the correlation coefficient will yield its minimum value ($c = -1$). Since p-electrode defects are subtle and most of the p-electrode will look similar to the reference image, the correlation coefficient will be in the range $0 \ll c < 1$.

Radial Statistics

Discontinuities of the p-electrode appear as grey-level variances along the ring, see Figure 6.2. Since we detected the correct centre of the p-electrode, we radially divide the



(a) Radial segments (b) Radial sampling (c) Image transformation

Figure 6.4.: (a) The p-electrode, compare Figure 6.2, is radially divided into $n = 8$ segments s_i, for which mean and standard deviation of pixels within each segment are computed. (b) The p-electrode is radially sampled using 32 directions α_i, i.e. angular resolution of 30 degrees; the intensities along each direction are computed by bilinear interpolation. (c) The p-electrode is sampled radially (counter clockwise) with an angular resolution of 1 degree and a spatial resolution of 1 px. In the transformed image, stretched to full range $[0, 255]$, the discontinuity is clearly visible as horizontal distortion in the lower part.

p-electrode into n segments s_i, compare Figure 6.4(a), for which we compute statistics of the segment's mean and standard deviation. Then, a defect might yield a mean intensity in one of the segments that is significantly larger than the mean of the other segments. Moreover, we can distinguish between a homogeneous or inhomogeneous texture of the p-electrode by analysing the standard deviation of the mean intensity and the intensity variations over all segments.

Alternatively, we analyse the p-electrode by radial sampling, where the p-electrode is sampled for different orientations α_i and the intensities are computed by bilinear interpolation (see Figure 6.4(b)). This image transform can also be interpreted as *unrolling* the image from the centre such that the new image can be inspected more easily even in case of manual inspection, see Figure 6.4(c). Furthermore, we can use the transformed image to obtain the features by computing the mean and standard deviation columnwise. A detailed description of our novel radial statistics can be found in Section 3.4.

6.2.3. Novelty Detection

In machine vision applications, often a receiver operating characteristic (ROC) is used to demonstrate the system's performance or to compare different models. A ROC analysis is based on the four outcomes that can be described by the confusion matrix C:

$$C = \begin{pmatrix} \text{true positive} & \text{false positive} \\ \text{false negative} & \text{true negative} \end{pmatrix} ,$$

with the definitions:

term	predicted label	true label
true positive	$+1$	$+1$
false positive	-1	$+1$
true negative	-1	-1
false negative	$+1$	-1

We use the methods described in Chapter 4 for novelty-detection: (i) kernel density estimator (KDE), (ii) kernel principal component analysis (KPCA), and (iii) OneClass-MaxMinOver (OMMO); then, the elements of the confusion matrix are evaluated by varying the discrimination thresholds, which are: (i) the estimated probability $p(\boldsymbol{x})$ for KDE, (ii) the reconstruction error $r(\boldsymbol{x})$ for KPCA, and (iii) the distance to the hyperplane in feature space, i.e. $d(\boldsymbol{x}) = \sum_i \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) - 1$, for OMMO. By using the elements of the confusion matrix we can visualise different ratios such as sensitivity and specificity; since we must obtain a system that can perfectly detect defects, we are interested in the ratio between true negative rate and false negative rate, and we will refer to these as detection rate (of defective samples) and false alarm rate (for defect-free samples).

6.3. Experiments

In many applications, model comparison or model selection based on ROC analysis is performed using the equal error rate (EER) or the area under curve (AUC), where EER is defined as the location on the ROC curve at which both detection rate and false alarm rate (FAR) have the same value. Hanley and McNeil [43] showed that the AUC can be interpreted as the probability that a randomly chosen negative sample is correctly classified or ranked with a greater suspicion than a randomly chosen positive sample; therefore the AUC is strongly connected to the nonparametric Wilcoxon statistic. However, EER and AUC are inappropriate measures if the application requires 100% detection rate for defects and minimum false alarm rate. Figure 6.5 compares two ROC scenarios, where minimising the EER or AUC does not necessarily lead to a minimum false alarm rate. We therefore evaluate the ROC curve at 100% detection rate and perform model selection by minimising the FAR.

We normalise each feature to $[-1, +1]$, and we perform 10-fold cross validation [93] to obtain the best parameters for each novelty-detection method, e.g. softness C and bandwidth σ in case of OMMO. Moreover, we randomly divided the dataset into 100 train and test sets (2/3 vs. 1/3) to evaluate the performance, and we apply a Wilcoxon sign rank test to analyse if the test errors are statistically significant. We compare the three novelty-detection methods with respect to three different feature sets: (i) Pearson's correlation coefficient, (ii) statistics of mean values for different number of radial segments ($s \in \{8, 16, 32, 64, 128, 256\}$), and (iii) statistics of mean values for

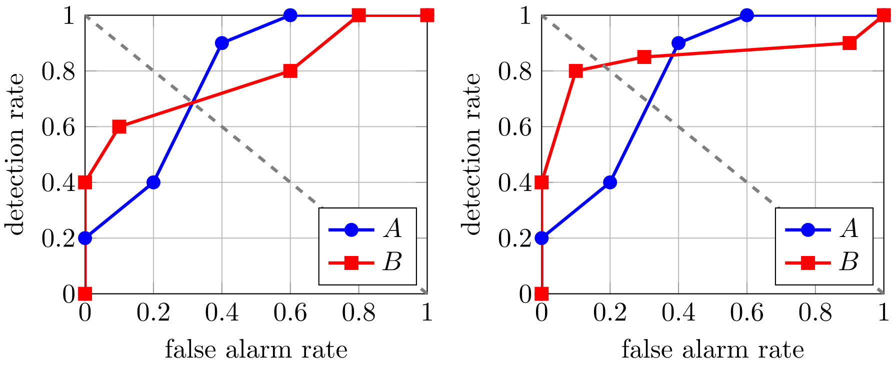


Figure 6.5.: Examples of ROC scenarios (the dashed line is the equal-error rate). Left: Two curves A and B with same EER and AUC but different false-alarm rate (FAR) at 100% detection rate. Right: Curve A has lower AUC and EER than curve B, but B yields a lower FAR at 100% detection rate.

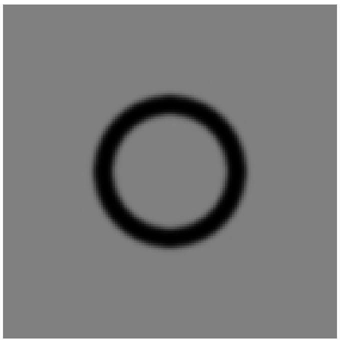


Figure 6.6.: Artificial image of a p-electrode used as reference for evaluating the correlation coefficient.

different number of sampled orientations ($\alpha \in \{8, 16, 32, 64, 128, 256\}$).

We cannot use the average defect-free p-electrode image as a template to compute the correlation coefficient, since the images contain different types of noise and we will produce a bias towards the most frequent noise. Instead of smoothing the images to reduce the noise, we created an artificial image of the p-electrode, see Figure 6.6, which serves as reference.

We have collected 53 images of LEDs, for which we extracted all 16 p-electrodes; each p-electrode was labelled by three experts, which results in 767 defect-free and 81 defective p-electrode images. Since some of defects are hardly visible (compare Figure 6.2), we treated a p-electrode as defective, if at least one expert recognised a defect.

6.4. Results

6.4.1. Correlation Coefficient

Figure 6.7 shows the ROC curve for the correlation coefficient; since we only tested one single feature in this scenario, an analysis of the ROC curve automatically leads to the best threshold for classification and therefore novelty detection is not required. On the one hand the correlation coefficient yields a good performance for EER (12%) and AUC (0.94), but on the other hand almost 90% defect-free samples are rejected, if we ensure 100% detection rate for defects. This indicates that the correlation coefficient can cover only large deviations and small defects cannot be detected, especially in the presence of strong image noise.

6.4.2. Statistics of Radially Sampled Orientations

Table 6.1(a) shows the results for radially subsampled features; the OMMO method outperforms the other methods for almost all values of α, and it yields the best FAR (0.65%) for $\alpha = 256$. The KPCA method yields superior performance compared to KDE, for $\alpha > 16$ and with a minimum FAR of 0.78%. The best performance of KDE is obtained for $\alpha = 256$ (1.3% FAR). The error rate of all novelty-detection approaches falls strongly below 5% FAR if α is increased from 16 to 32 orientations and it drops further by almost a factor of two for $32 < \alpha < 256$, which indicates that there are several small defects.

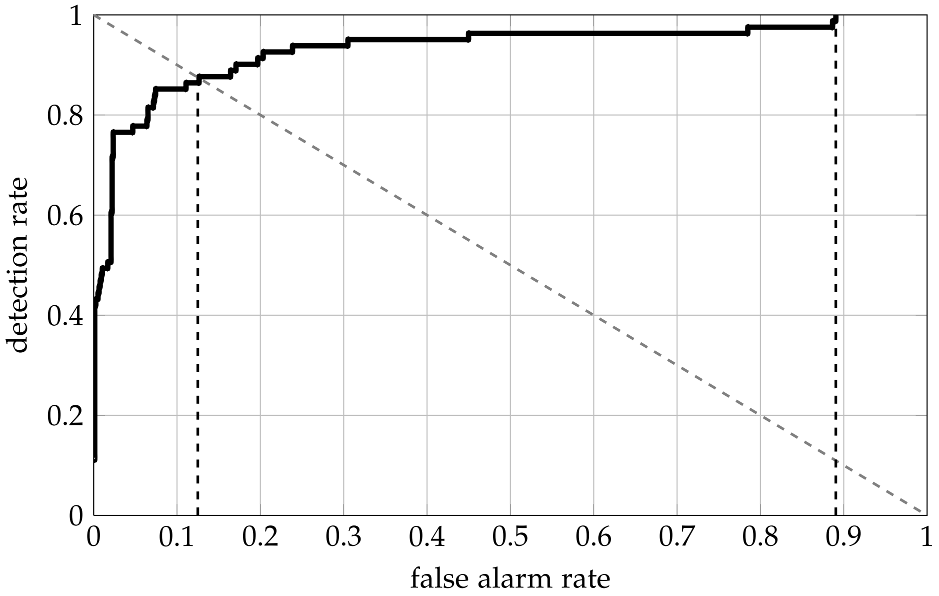


Figure 6.7.: ROC curve for Pearson's correlation coefficient. The values for EER (12%) and AUC (0.94) are acceptable, whereas the minimum FAR at 100% detection rate is very high (0.89%).

Table 6.1.: False alarm rate at 100% detection rate using the two approaches for computing radial statistics; the mean FAR is computed over 100 test sets.

(a) Results using radial subsampling

number of	mean FAR [%]		
orientations α	KDE	KPCA	OMMO
8	66.9	68.8	63.8
16	70.6	83.3	54.7
32	4.2	3.6	3.9
64	2.6	1.4	1.8
128	1.4	0.91	0.76
256	**1.3**	**0.78**	**0.65**

(b) Results using radial segments

number of	mean FAR [%]		
segments	KDE	KPCA	OMMO
8	6.2	7.0	7.8
16	5.6	3.2	3.5
32	4.1	4.2	3.9
64	1.8	1.8	1.0
128	1.2	0.9	0.51
256	**1.1**	**0.81**	**0.13**

Since the p-electrode is sampled radially using our feature-extraction methods, a large number of orientations is required to also sample the defective regions. With few orientations, e.g. $\alpha = 16$, several defects are located in between two orientations, which explains the FAR of over 70%. Since the sampling technique performs an oversampling for more than 140 orientations as described in Section 3.4, the FAR improves only slightly when using 256 orientations instead of 128.

6.4.3. Statistics of Radial Segments

Table 6.1(b) shows the results for radially encoded segment features. Similar to the previous feature set the OMMO approach yields the best performance by far, 0.13% FAR, for a large number of segments. In contrast to the previous feature set, KPCA outperforms KDE only in few cases, i.e. for 16, 128 and 256 segments. Already for few segments, e.g. 8, the FAR of all approaches is below 8% and drops further below 2% for 64 segments.

Both feature sets yield almost the same performance (3.6% – 4.2%) for 32 orientations and 32 segments independent of the novelty-detection method; for more than 32 segments and orientations, the segment-based features yield significantly lower false alarm rates. Since a single segment contains more information in terms of pixels than a single sampled radial orientation, the segment statistics are less prone to image noise and hence outperform the statistics of sampled orientations.

6.4.4. Comparison

The results indicate that features encoded by using radial segment statistics can capture p-electrode defects more accurately than radially-sampled orientations, especially in the presence of image noise. Moreover, the OMMO approach outperforms both KPCA

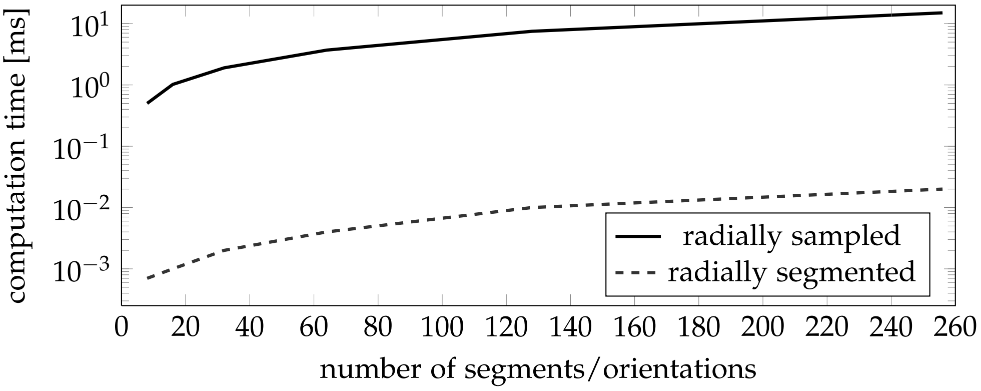


Figure 6.8.: Comparison of the computation time per image of both feature-extraction methods.

and KDE with an overall performance of 0.13% false alarm rate at 100% detection rate.

Figure 6.8 compares the computation times for both feature-extraction methods; a small number of orientations yields a computation time of less than 1ms, whereas up to 10ms are required for many orientations; the computation time grows linearly with the number of orientations.

The computation times of radial-segment statistics follow the same rule as for the radial sampled orientations except a scaling factor of approximately 750, which is due to the bilinear interpolation that we must apply to compute the intensities on each orientation. However, this scaling factor may decrease once a faster bilinear interpolation technique is applied.

6.5. Experiments and Results for Centre Localisation

In Section 2.2 we have proposed two novel approaches to accurately localise the centre of a (semi-) circular object, and we have demonstrated, based on synthetic images, that our novel approaches do not only significantly outperform common approaches, but also provide accurate results in case of strong image noise.

In this section, we will show that our novel approaches yield high performance for accurate localisation of the p-electrode's centre. Similar to the experiments for synthetic images, we compare with the approach that employs a Hough transform.

We have selected the 82 most difficult images, for which the centre of the p-electrode was labelled by experts. We applied the centre localisation approaches with the same parameter settings mentioned in Section 2.2, and we evaluated the Euclidean distance between the correct centre and the estimated centre.

Table 6.2 shows the quantitative results for centre localisation of p-electrodes; similar to the results for synthetic images, the symmetry-based approach significantly outper-

Table 6.2: Results for the three approaches for centre localisation applied to images of p-electrodes. The accuracy is evaluated by the Euclidean distance between estimated and correct centre over 82 images, which were labelled by experts.

method	mean error (std.)
Hough transform	3.97 (5.65)
gradient-based	2.16 (2.52)
symmetry-based	**0.98 (1.16)**

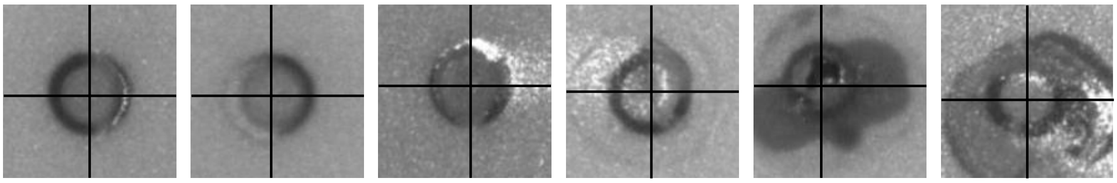


Figure 6.9.: Example images for which the centre has been detected by using our novel symmetry-based approach, which is extremely robust to noise, occlusions, and reflections.

forms the other approaches with a mean error of only 0.98 pixel, whereas the gradient approach yields an error of 2.16 pixel. Since some of the p-electrodes are partially occluded and affected by strong noise, the contour obtained by an edge detector does not correspond to the contour of the p-electrode; thus, the estimated centre of the Hough transform is inaccurate for most p-electrode images.

Figure 6.9 shows the qualitative results of the symmetry-based approach; we can observe that the correct centre is successfully detected for almost all images and even in cases of higher image degradations such as image noise in combination with occlusions and strong reflections.

6.6. Discussion

We have demonstrated that p-electrodes in LED images can successfully be inspected by a machine vision systems that employs novel methods for preprocessing, feature extraction and novelty detection. The experiments were performed on a dataset of 848 p-electrode images, which consists of 767 defect-free and 81 defective images.

First, we have applied two novel approaches to detect the p-electrode's centre with high accuracy. The first approach employs image gradients from which the centre is estimated as the location where most gradient vectors intersect. The second approach evaluates the radial symmetry of grey levels and the location where the mean grey-level variation reaches its minimum. Both approaches are robust to occlusion and strong noise for images of p-electrodes, while preserving simplicity and efficiency. The symmetry-based approach yields the best accuracy with an error of less than one pixel, averaged over the 82 most difficult images of p-electrodes.

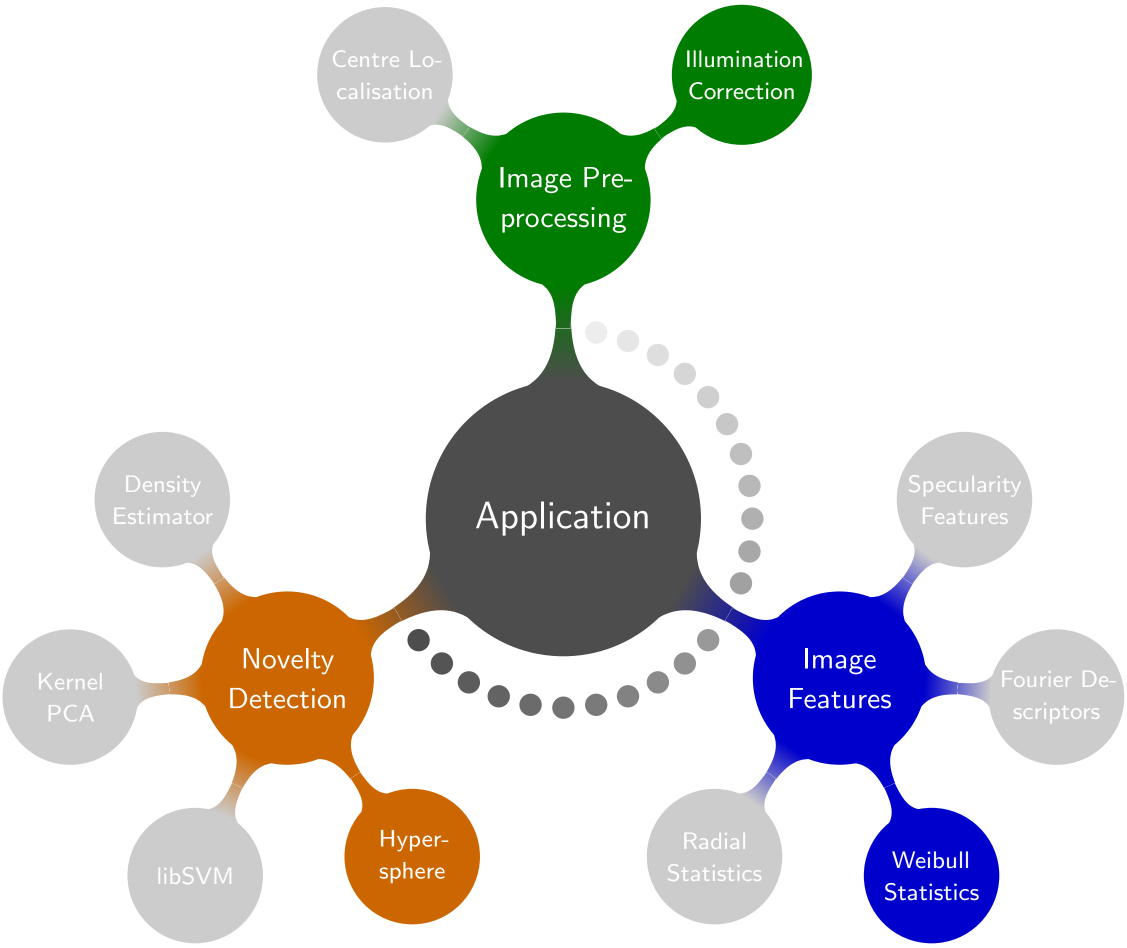
Second, we have computed the correlation coefficient and two novel methods that capture radial statistics; one method divides the p-electrode into radial segments for which we compute the mean and standard deviation, the other method performs a sampling on different orientations to evaluate mean and standard deviation.

Third, we have applied different novelty-detection methods instead of commonly used two-class classification methods, since the underlying problem is the detection of atypical/defective samples. In particular we use three novelty-detection methods: kernel density estimation (KDE), kernel principal component analysis (KPCA), and a support-vector data-description method (OMMO). Since the inspection system must detect every defect, we perform model comparison by determining the minimum false alarm rate at 100% correct detection rate for the ROC analysis.

Since the images contain strong noise and the defects are small, the correlation coefficient leads to a high false alarm rate of almost 90%. Moreover, we could observe that the radial-segment features not only outperform the radial-sampling features in terms of false alarm rate, but also in terms of efficiency. It turned out that OMMO achieves the best overall performance by far with an false alarm rate of only 0.13%, which is even better than human performance. In contrast, the other novelty-detection methods yield false alarm rates of 0.8% (KPCA) and 1.1% (KDE).

The proposed feature-extraction methods are simple and the resulting feature space is low-dimensional (10 features). Hence, they can also be used for other (real-time) industrial applications such as surface inspection (if the relevant objects are ringlike or circular), iris classification or cell classification. Since the LED images contain strong noise and only 81 defective samples were available, the performance may further improve once more images are available.

7. Defect Detection in Texture Images



This chapter demonstrates the application of local Weibull statistics to the problem of defect detection in texture images. The data has been provided by Robert Bosch GmbH, Stuttgart. Some of the work described in this chapter has been previously published in [101].

7.1. Introduction

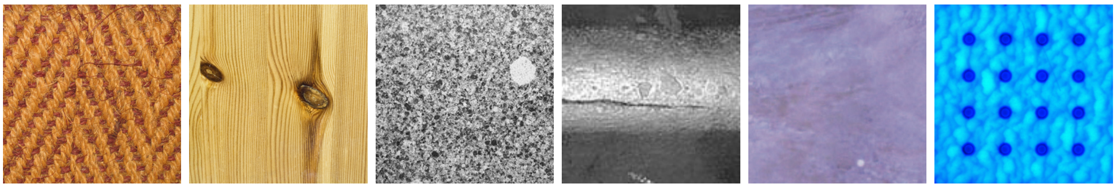
Over the past decade, automated optical inspection has proven to reduce the cost of industrial quality control significantly. Automated defect detection in textured surfaces has increasingly gained importance in industrial production. One of the major applications of texture defect detection is surface inspection such as the inspection of semi-conductor components or textiles. Even though automated inspection systems have advantages over human inspection such as reliability and reproducibility, they are often highly adapted to a particular set of defects and they fail if the problem changes and new types of defects arise.

Texture defects can be either non-textured areas or areas that locally differ from the background texture of the surface. These defects are often subtle and very hard to identify even manually; moreover, well-defined defect specifications and pixel-wise defect labelling are usually unavailable. Figure 7.1 shows some example surfaces such as textile, wood, or steel with different types of defects.

Several approaches for texture defect detection have been proposed; generally, they can be separated into two categories: local and global approaches. Since global approaches are applied to the whole image, they yield accurate results for defects that affect the overall appearance such as shade or tonality, for instance, but they perform poorly in texture defect detection. Therefore, approaches that evaluate local texture characteristics are mostly applied to detect small defects in textures. These approaches can be separated into statistical approaches (e.g. grey level statistics, fractal dimension), spectral approaches (e.g. Gabor filter, Fourier analysis), and model-based approaches (e.g. Markov random fields, model-based clustering). However, these approaches are often too complex, adapted to a particular defect type, and fail to detect miscellaneous types of defects.

Scholte et al. recently reported that brain responses strongly correlate with Weibull image statistics when processing natural images [91]; other researchers found that Weibull parameters estimated from the distribution of magnitudes of image gradients yield accurate results for texture classification [116, 9, 37, 36]. However, Weibull parameters have not been applied to the problem of texture defect detection.

Here, we propose a simple, non-parametric machine vision system for texture defect detection based on the Weibull features we have introduced in Section 3.5. We review the steps involved in our novel defect detection approach in Figure 7.2. For each local image patch we obtain a two-dimensional feature vector that contains the estimated shape and scale parameter of the corresponding Weibull distribution. Based on these two Weibull features we apply a novelty-detection method, which evaluates the Euclidean distance of each local patch to the median in the Weibull space. We assume that most of the defect-free patches can be described by a single cluster and every defective patch will significantly deviate from this cluster region. Hence, if the maximum distance is



(a) textile (b) wood (c) stone (d) steel (e) ceramic (f) LED

Figure 7.1.: Example textured surfaces with different types of (subtle) defects. Note that some defects may only be visible in the electronic version.

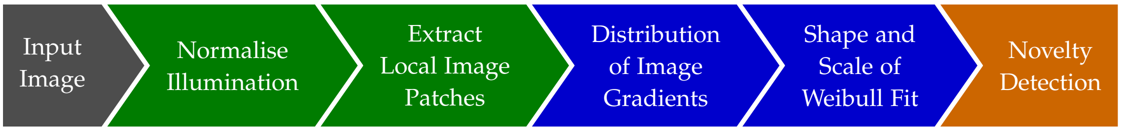

Input Image
Normalise Illumination
Extract Local Image Patches
Distribution of Image Gradients
Shape and Scale of Weibull Fit
Novelty Detection

Figure 7.2.: Scheme of our novel system for detecting arbitrary defects in texture images.

larger than a threshold, the current image patch is considered as defective (negative); the threshold is automatically determined by minimising a cost-function. Compared to existing approaches, our approach is more efficient by orders of magnitude and does not involve any kind of learning or further parameters.

We evaluate the performance of our method by using the highly challenging database of the contest *weakly supervised learning for industrial optical inspection*, which was introduced at the DAGM conference 2007. The images were proposed by the company Robert Bosch GmbH and consist of different texture classes and different defect types. We demonstrate that our method is able to detect arbitrary deviations from the reference (background) texture.

Detecting defects in textures is a very challenging problem in computer vision. Although several methods for texture defect detection have been proposed, recent reviews [115, 61] have shown that there is a clear need for standard evaluation and for standard benchmark datasets in order to avoid highly adapted methods. Supporting these aspects, we define the requirements for texture defect detection as follows:

1 Not only a single defect type needs to be detected, but arbitrary defects must be detected.

2 The method must not be adapted to a specific background texture, instead it must perform well on various background textures.

3 Specifications of defects are missing and the defects are weakly labelled (not pixel-wise).

4 Class-dependent costs must be incorporated (samples with missed defects are worse than defect-free samples classified as defective).

5 All parameters, e.g. for filtering, feature extraction, or learning, must be determined automatically.

In addition to the above mentioned requirements, our defect detection system has the following property:

6 No exhaustive training set is required—our system works even if only a single defective image is available.

Most approaches proposed for texture defect detection meet only few of these requirements—we present a machine vision system that meets all requirements, that yields accurate results, and that is extremely efficient.

7.2. Feature Extraction

We will apply our feature extraction approach proposed in Section 3.5, which describes local gradient distributions using only two features: shape and scale parameter of a Weibull fit; these parameters are computed for gradient magnitude distributions of local image regions.

7.3. Novelty Detection

In Chapter 4 we have described several state of the art methods for novelty detection, and we have proposed a novel and simple approach for computing a data description solely based on support vectors. Even though these methods yield superior results in several benchmarks, we are interested in the simplest model that detects novel samples and can also be applied even if only a single image of the texture class is available.

Figure 7.3 depicts a straightforward method for novelty detection, which involves the computation of a hypersphere and can be described as: (i) compute a reference point such as mean, centre of mass, or median, (ii) determine a threshold for the maximum distance of a defect-free sample to the reference point. Even though this method is simple, it yields accurate results if the samples are symmetrically distributed around the reference. Furthermore, there is only one free parameter, which can be determined automatically by analysing the variance of defect-free samples or by minimising the classification error if outlier samples are available. For this work, the reference point is represented by the median, since few outliers (defective samples) might have a strong influence on the mean, which will result in an unstable model. This simple novelty-detection method provides two major advantages: We are no longer restricted

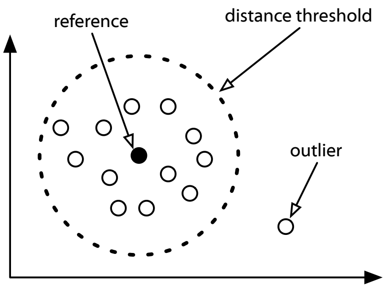


Figure 7.3: Novelty detection based on the analysis of distances to a reference point; samples are rejected if their distance to the reference is larger than the threshold. The reference can be mean or median of all samples of a defect-free image, for instance; the threshold can be computed by using a single defective image.

to a particular class of textures or class of defects and we can perform defect detection for each image individually.

The proposed novelty detection method is closely related to the support-vector data-description (SVDD) approach by Tax et al. [96, 97], where a hypersphere is computed such that the volume is minimised and the centre is described by support vectors. However, we generally have to optimise two parameters for SVDD, the softness parameter as well as the distance threshold. Moreover, if new samples become available retraining can be more time consuming compared to the proposed approach.

7.4. Experiments and Results

We use four classes of texture images provided by the company Robert Bosch GmbH for the DAGM 2007 contest on weakly supervised learning of texture images. The contest made one of the most challenging databases for texture defect detection available; this is proven by the fact that only the contest winner obtained acceptable results. Unfortunately, explicit information about the winning method as well as a detailed result evaluation has not been published so far. We will elaborate our novel method and discuss the results to further promote this challenging dataset.

The texture classes we use in this work can be described in the following way (see Figure 7.4):

Class 1 Defective regions are rather homogeneous with intermediate grey levels and elliptic shape, whereas the defect-free background texture contains high-frequency structures with grey levels of high contrast.

Class 2 Defective regions are bright and with fractal shape, whereas the background contains a speckle pattern of medium size and a global linear gradient with different orientations.

Class 3 Defective regions contain grating-like structures of varying contrast, whereas the background consists of speckle pattern and globally varying local contrast.

Class 4 Defective regions are dark and with elongated shape, whereas the background contains large speckles with high contrast.

For each texture class the defect size is not constant, but varies to a limited but not specified extent. Furthermore, the defects are only weakly labelled by a surrounding ellipse, i.e. pixel-wise defect locations are unavailable and the labelled region also includes defect-free areas. Each texture class consists of 1000 defect-free and 150 defective greyscale images (512×512 pixel).

7.4.1. Experiment Settings

We have to determine three parameters for the novel texture defect detection approach: (i) size of the local image patches, (ii) width of Gaussian derivative filter, and (iii) distance threshold for novelty detection. Even though defects are only weakly labelled, we use the defect statistics to automatically set the patch size. Therefore, we compute the minimum minor axis of all defective regions and round it to the next power of 2 for computational reasons, which yields a patch size of 32×32 pixel for all classes. Furthermore, we must define the overlap of local patches to avoid border artefacts. Unfortunately, the number of image patches grows exponentially with increasing the overlap, which becomes computationally intractable for real-time applications. For a compromise, we use an overlap of 50% but slight changes of the overlap do not significantly change the results.

Moreover, we apply directional Gaussian derivative filters to the image to obtain the gradient magnitudes. The size w and the standard deviation σ of these Gaussian filters are chosen according to the patch size p, where we use the following relationships $w = \text{ceil}(p/11)$ and $\sigma = w/5$. These relationships were experimentally evaluated, but slight changes will yield similar results.

Since missing a defect (false positive) is worse than detecting a defect for a defect-free sample (false negative), we employ asymmetric costs for classification errors. According

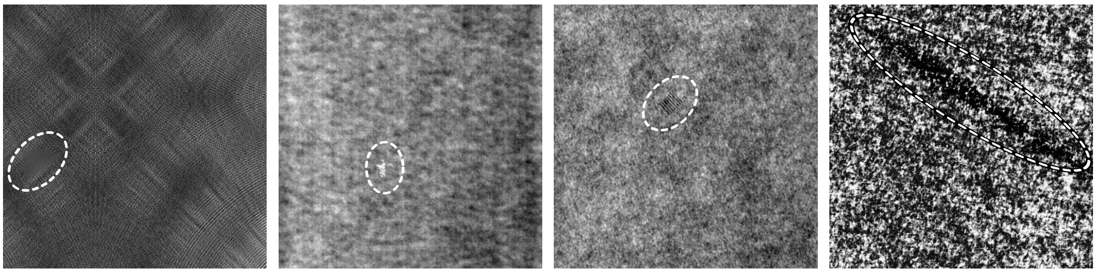


Figure 7.4.: Example images for the classes of textures used for the experiments. Defects are weakly labelled by ellipses and contain also defect-free regions. The four texture classes significantly vary concerning the background texture as well as concerning the defect characteristics. Whereas defects for the first three texture classes are small, the defects of the fourth class have a large extent and cover larger image regions.

to the terms of the contest, we use the following costs: (i) false positives are penalised with 20 and (ii) false negatives are penalised with 1. The novelty detection is performed such that the texture image is classified as defective if the maximum distance to the median in the Weibull space of all patches is larger than a threshold, and we optimise this threshold by minimising the total costs. Since our novel defect detection method does not involve any kind of learning scheme, we use all 1150 images of each texture class for estimating the classification error.

7.4.2. Results & Discussion

We have already discussed that Weibull image statistics can be used to classify visual content (see Section 3.5), but how do they perform for measuring local texture deviations?

Figure 7.5 shows an example image of the first texture class; the Weibull fit of local patches in defective region yields shape < 1.9 and scale > 17, whereas patches in defect-free regions have shape > 2 and scale < 14. This indicates that the Weibull features can successfully describe local texture deviations.

Figure 7.6 depicts the distribution of Weibull features of all local patches within a defective image; the four patches with maximum distance to the median are detected, and its location within the image is highlighted. Apparently, defect-free patches build a single cluster and defective patches can easily be identified as outliers; this holds for all four texture classes. However, if the pattern of the background texture shows large variations, some defect-free patches also have a larger distance to the median (see Figure 7.6 first and second column).

Moreover, we evaluate also receiver operator characteristics (ROCs) for all texture classes to demonstrate the performance for different distance thresholds (see Figure 7.7 and Table 7.1). Based on the ROC we compute the equal-error rate (EER), which represents the optimised error rate for which positive and negative errors are equally important, and we compute the area under the curve (AUC) of the ROC, which corresponds to the overall performance independent of a particular threshold. Furthermore, we apply asymmetric weights as described before to optimise the total cost (TC).

We obtain an EER of 8.5%, AUC of 0.96, and a total cost of 190 for the first texture class, although the combination of defect type and background pattern is very challenging—some defects are hard to detect even manually (see Figure 7.4). The results for the second and third texture class are almost perfect regarding EER (0.1%, 1.3%) and AUC (0.99 for both) and also the total costs are very low (2 and 28). Our inspection system yields high accuracy (EER 3.2%, AUC 0.99, TC 51) for the fourth class, even though this class is very different from the other classes. The results prove that our inspection system can successfully deal with different challenging defect types on varying background textures and yield accurate results in detecting defects in texture images.

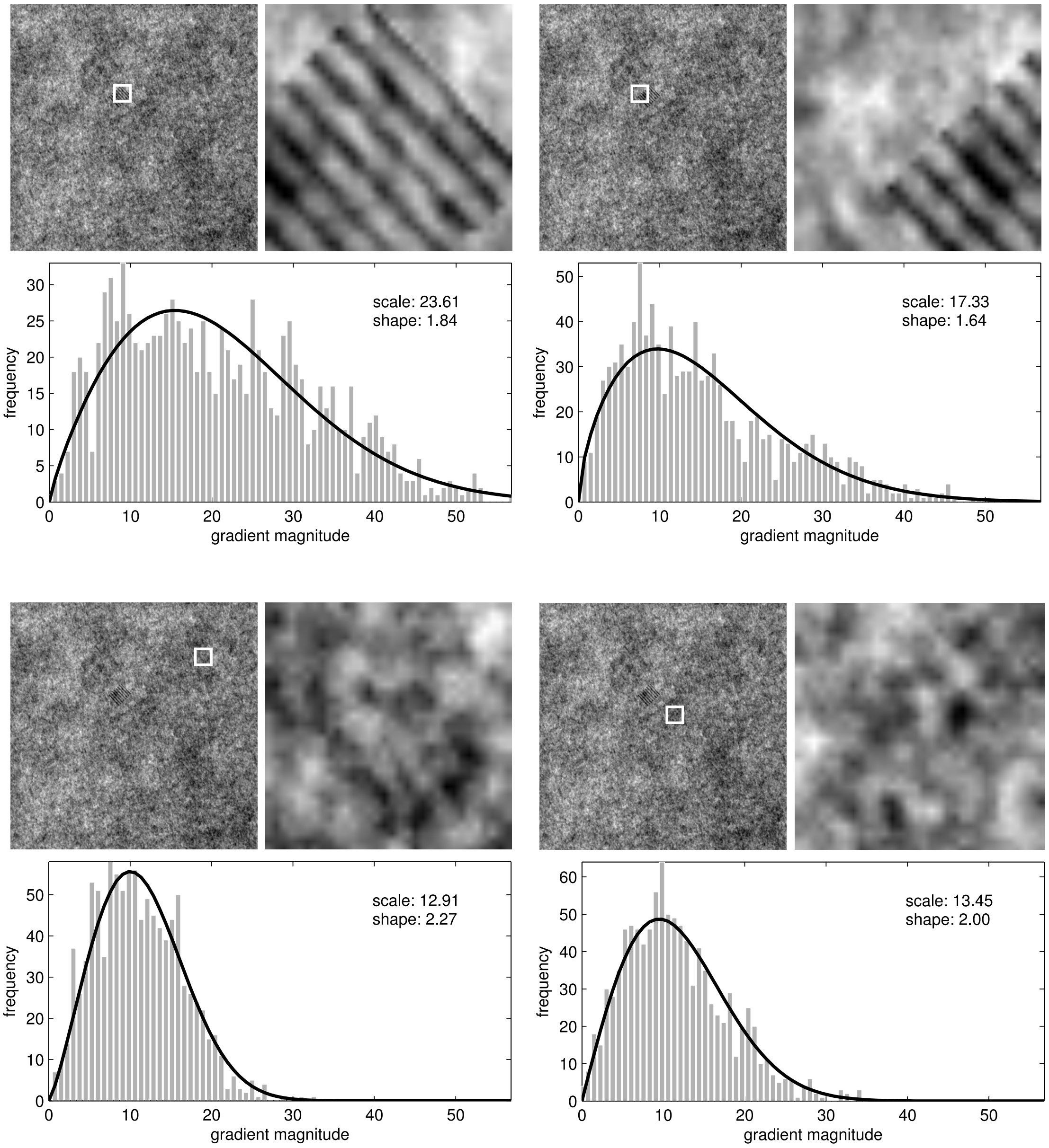


Figure 7.5.: Example image of the first texture class containing a (grating) defect. The first and second row show two local patches at the defective region and their Weibull fit to the distribution of gradient magnitudes. The third and fourth row depict two defect-free patches and the corresponding Weibull fit. Obviously, the distributions of gradient magnitudes for defective and defect-free patches differ and hence the Weibull features shape and scale yield significantly different values.

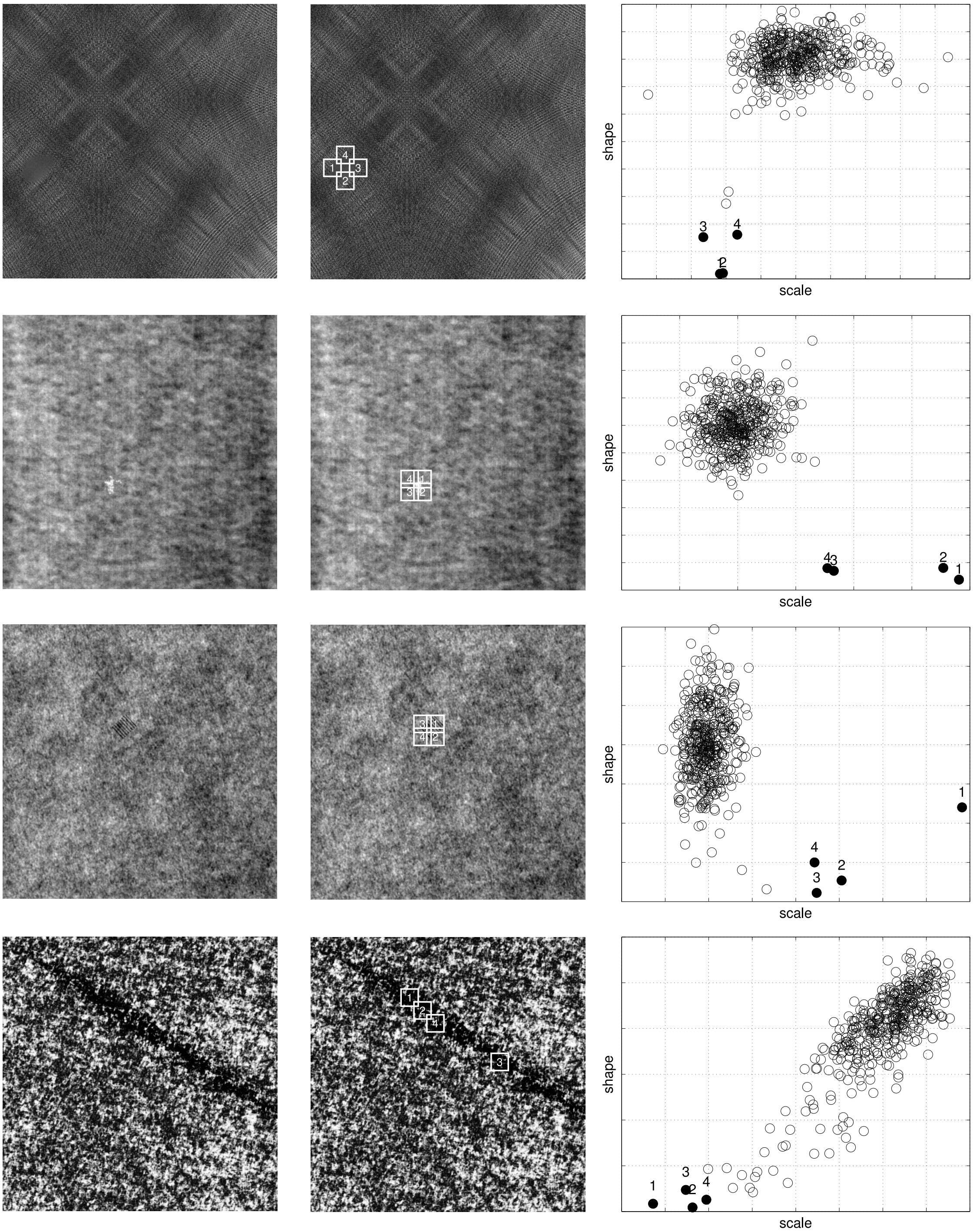
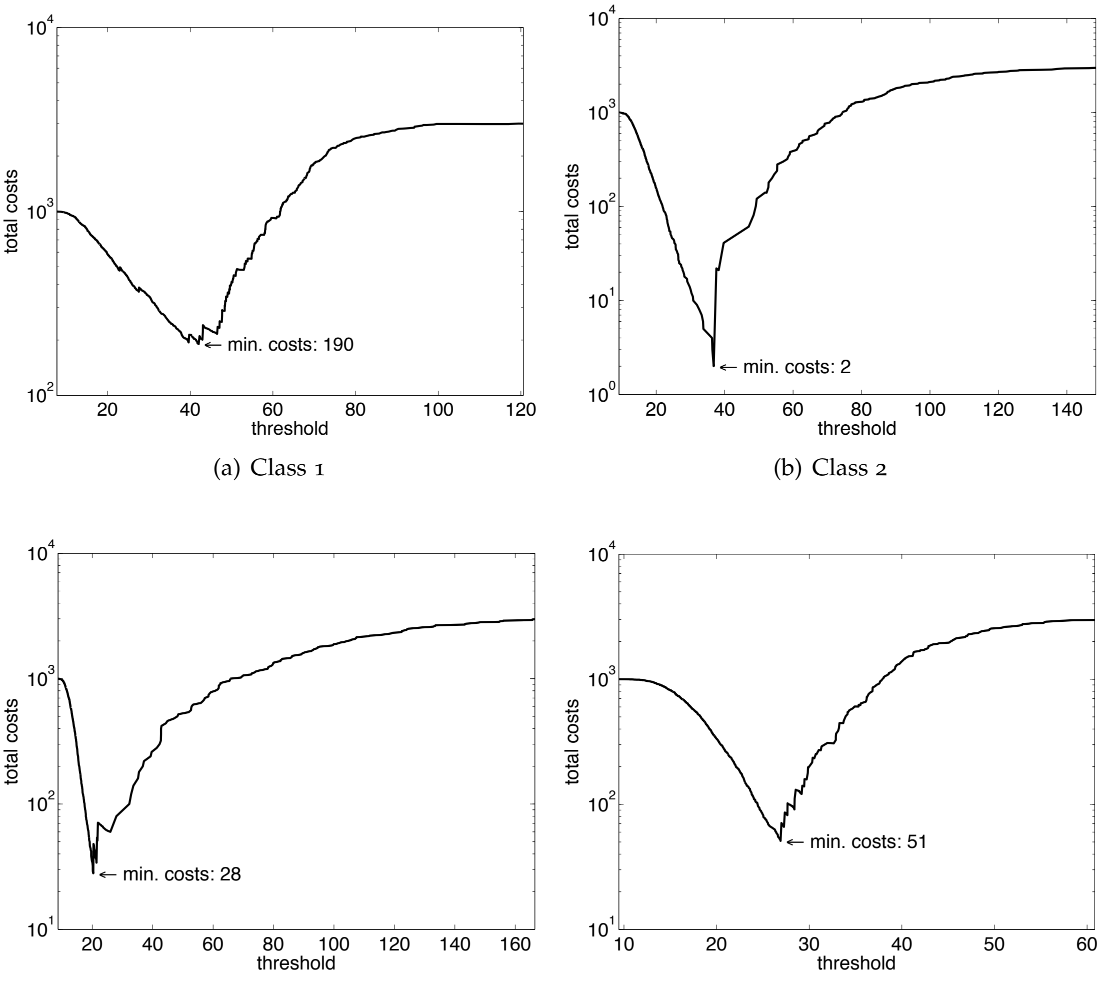


Figure 7.6.: Demonstrating the usefulness of local Weibull parameters for defect detection. One example defective image for each texture class is shown (first column)—compare with Figure 7.4 for defect description and defect location. The distribution of Weibull features for all local patches (third column) forms a cluster such that outliers can be identified by analysing the distance to the median. The four samples with maximum distance to the median are depicted in the Weibull space (third column) as well as their corresponding location in the image (second column).



(a) Class 1

(b) Class 2

(c) Class 3

(d) Class 4

Figure 7.7.: Total costs for varied distance thresholds of each texture class. For the best thresholds the total costs are between 2 and 190, which means that 2 out of 1000 defect-free images are classified incorrectly in the case of class 2.

7.5. Discussion

We have described a novel machine vision system for defect detection in texture images. Our system evaluates the distribution of local gradient magnitudes based on a Weibull fit; we have used the two Weibull parameters to perform simple novelty detection; we have computed the median of all samples in the Weibull space and rejected an image if the maximum distance to the median is larger than a threshold.

We have extensively evaluated the performance of our novel system on the very

Table 7.1.: Performance evaluation of the novel defect detection approach applied to four texture classes. Based on the receiver operator characteristic the equal error rate (EER) corresponds to the error rate at which positive and negative samples are weighted identically, whereas the area under the curve (AUC) captures the overall performance for every possible class weighting. By optimising the distance threshold such that the sum of false positives (FP) and false negatives (FN) reaches its minimum, we retrieve the total costs (TC). Note that for optimising total costs asymmetric weights have been applied here, false negatives count 1 and false positives count 20. The false negative rate (FN*) at 100% true negative rate indicates the amount of defect-free images where a defect has been detected, while every defective image has been correctly classified.

Class	EER	AUC	TC	FP	FN	FN*
1	8.5%	0.96	190	3	130	47.0%
2	0.1%	0.99	2	0	2	0.2%
3	1.3%	0.99	28	0	28	2.8%
4	3.2%	0.99	51	0	51	5.1%

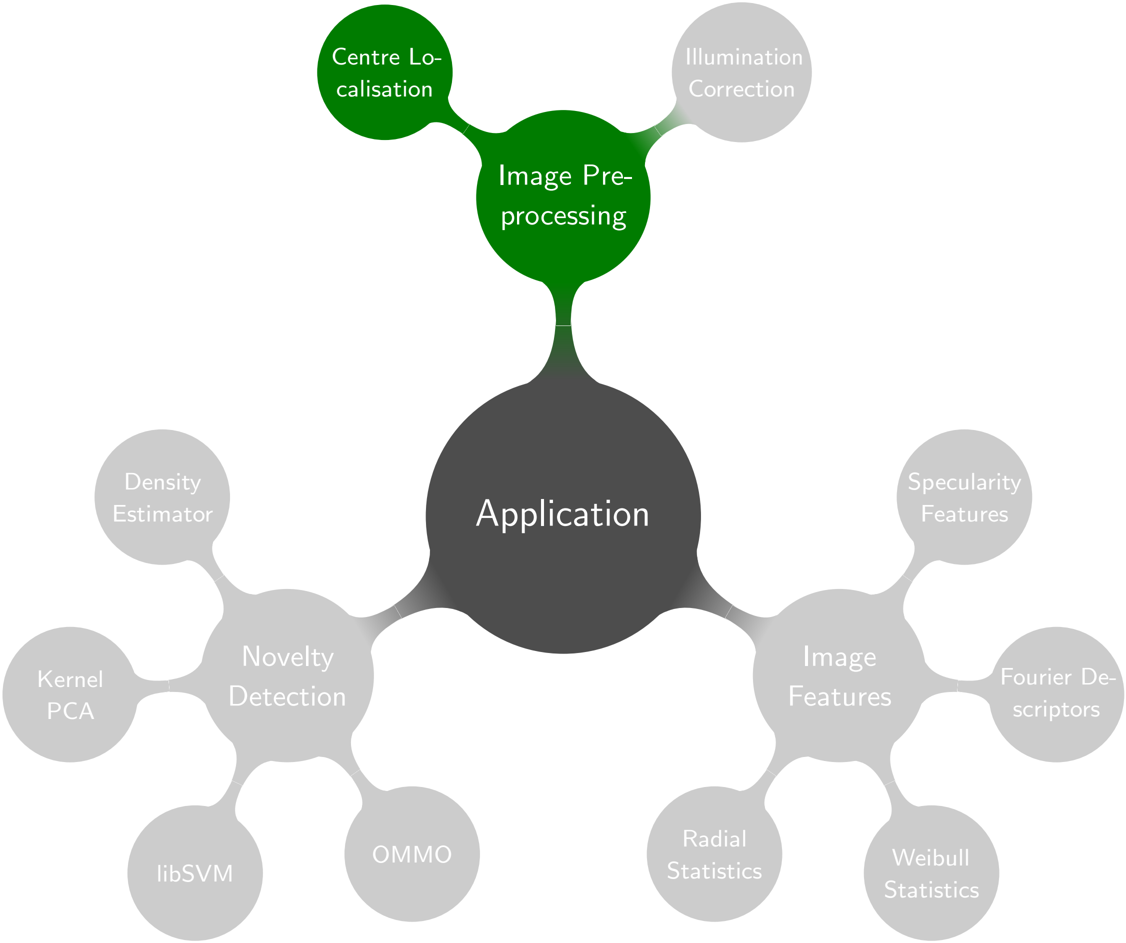
challenging dataset of the DAGM 2007 contest. Our system has yield accurate results for all texture classes, whereas two classes have been classified with an equal error rate of less than 1.3%. Even in case of large variations of the background pattern within each texture class and in case of subtle defects, our novel system has proven to be robust and has yield high accuracy. We have observed that our system can successfully deal with complex textures and it detects even small and subtle deviations. Additionally, we have not employed any sophisticated learning algorithms; therefore, we can omit expensive re-training and exhaustive parameter optimisation. Due to its efficiency our approach can even be applied to real-time applications. Only the winning team of the DAGM 2007 contest achieved acceptable results and met the maximum allowed total costs of 200 for each dataset; unfortunately, neither the error rates nor the methods of the winner have been published.

Since the extracted features are already very powerful in the sense that they form distinct clusters, we do not expect a significant improvement if more sophisticated methods for novelty detection are applied. However, for other feature sets the novelty-detection methods proposed in Chapter 4 are more powerful than the simple hypersphere we have employed here.

Humans can recognise arbitrary defects in textures almost immediately, since these regions often seem to stand out compared to the background and therefore defects can also be interpreted as regions with significant saliency. Several approaches have been proposed in human vision science to detect and locate salient image regions, and we believe that these methods could also be applied to the problem of texture defect detection. Recently, methods for the detection of salient image points employed sparse coding techniques to model the subspace of these regions. In contrast, we could try to model or learn the subspace of all defect-free regions in a texture image by using sparse coding techniques, for example; with such a sparse representation a defective region should not be recovered as accurately as a defect-free region.

In general, we believe that every method for texture defect detection should be evaluated on standard benchmark datasets and we hope for the future that the dataset we have used here will also be used more frequently by other researchers.

8. Accurate Eye Centre Localisation



This chapter demonstrates the application of gradient-based centre localisation to the problem of eye centre localisation. Some of the work described in this chapter has been previously published in [99]; a demonstration can be found at `http://www.youtube.com/watch?v=aGmGyFLQAFM`.

8.1. Introduction

The localisation of eye centres has significant importance in many computer vision applications such as human-computer interaction, face recognition, face matching, user attention or gaze estimation [11]. There are several techniques for eye-centre localisation, some of them make use of a head-mounted device, others utilise a chin rest to limit head movements; note that eye-centre detection is not an eye tracker yet. Moreover, active infrared illumination is used to estimate the eye centres accurately through corneal reflections. Although these techniques allow for very accurate predictions of the eye centres and are often employed in commercial eye-gaze trackers, they are uncomfortable and less robust in daylight applications and outdoor scenarios. Therefore, available-light methods for eye-centre detection have been proposed; they can roughly be divided into three groups: (i) feature-based methods, (ii) model-based methods, and (iii) hybrid methods.

Feature-based methods make use of eye properties such as symmetry, shape, or colour to estimate the eye centre locations. In the case of grey level images, one of the simplest methods is to take the position of the darkest pixel as eye centre estimate; however, this method will be very sensitive to image noise, especially in low resolution images. More robust methods employ edges, corners, or image gradients to locate the eye centres. Usually, feature-based methods are applied at the end of a multi-stage scheme, which consists of: (1) face detection, (2) coarse estimation of eye regions, and (3) fine estimation of eye centres. Since face detection methods have become very effective and accurate [111] and rough eye regions can be extracted easily, such a multi-stage approach is computationally efficient and can therefore be integrated into systems with low computation performance such as mobile phones or other portable devices.

Model-based methods make use of the holistic appearance of the eye including eyelid, iris, pupil, or eyebrow. After the eye model has been learned from a set of training images, it is matched to unknown faces for estimating the eye locations. For learning, a set of features at different locations within the face is computed, e.g. Haar-like features, wavelet coefficients, or texture features. Since not only the pupil is considered but also several other structures around the pupil such as iris, eyelid, or eyebrow, model-based methods achieve more robust results in detecting the overall eye locations compared to feature-based methods. However, accurate estimation of the eye centres is often disregarded and the eye centres are estimated as being in the middle of the eye corners or in the centre of the eye model. Therefore, these methods achieve poor results regarding accuracy. Furthermore, model-based methods involving any kind of learning or classification scheme have several drawbacks, e.g. retraining has to be performed if the data changes, computational complexity can be high, or model parameters must be tuned.

Hybrid methods are combinations of a model-based and a feature-based method,

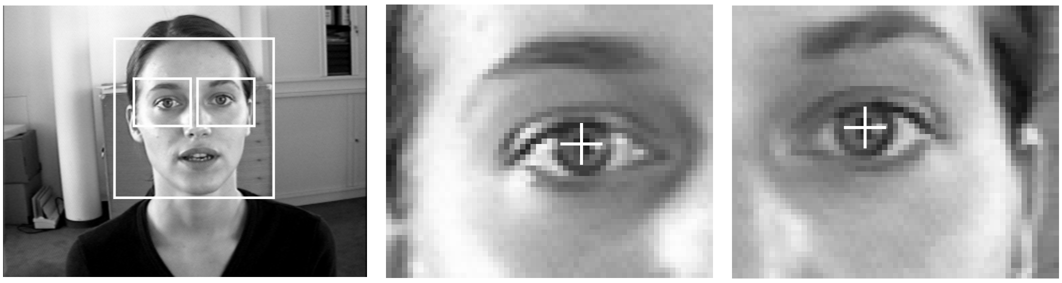


Figure 8.1.: Multi-stage approach for eye centre localisation: A face detector is applied and based on the face location rough eye regions are extracted (left), which are used for a fine estimation of each eye centre afterwards (middle and right).

where candidate eye regions are determined first by some feature descriptor or learnt eye model and then used by one or more classification frameworks to identify the correct eye centres. Even though these methods often yield the best performance due to their robustness, they cannot be easily integrated into other systems due to their computational complexity.

Instead of elaborating in detail all methods that have been proposed for eye centre localisation, we refer to Table 8.1, where a number of methods and their properties are listed. We here propose a feature-based approach for eye centre localisation that can efficiently and accurately locate and track eye centres in low resolution images and videos, e.g. in videos taken by a webcam. We follow the multi-stage scheme that is usually performed for feature-based eye centre localisation (see Figure 8.1), with the following steps: (i) we apply our novel gradient-based approach for accurate centre localisation, proposed in Section 2.2.2, which defines the centre of a (semi-)circular pattern as the location, where most of the image gradients will intersect, and which can be computed efficiently, (ii) we incorporate prior knowledge about the eye appearance to eliminate local maxima, reinforce centre estimates and increase robustness, and (iii) we apply simple postprocessing techniques to reduce problems that arise due to glasses, reflections inside glasses, or prominent eyebrows. Furthermore, we evaluate the accuracy and robustness of the proposed approach to changes in lighting, contrast, and background by using the challenging BioID database[1]. The obtained results are extensively compared the with state-of-the-art methods for eye centre localisation presented in Table 8.1.

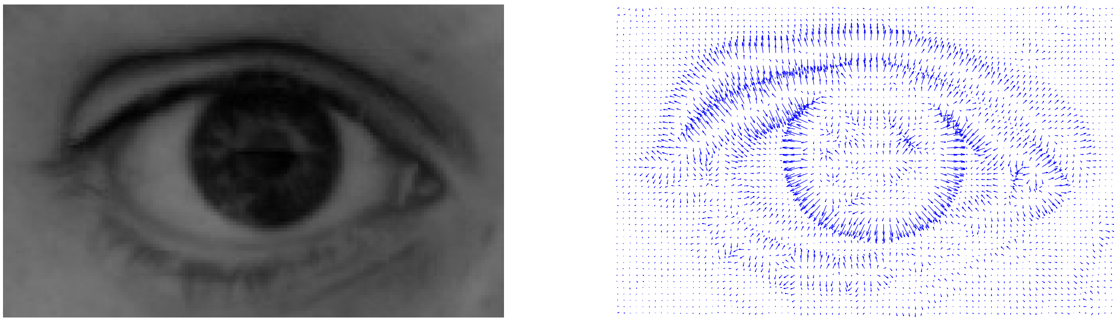
8.2. Methods

We briefly motivate our gradient-based approach for centre localisation in case of detecting the pupil's centre. Geometrically, the centre of a circular object can be detected

[1] http://www.bioid.com/support/downloads/software.html

Table 8.1.: Comparison of some recent methods for eye centre localisation. Whereas early approaches mostly employed simple features such as edges or grey value and sophisticated learning schemes, in 2006 many methods that compute wavelet-based features and use complex multi-stage classifiers were proposed. More recently, combinations of simple image features and common classifiers such as grey values and linear discriminant analysis (LDA) have been used.

Year	Reference	Involves Learning?	Features	Model / Learning Scheme	Preprocessing
2010	[6]	–	Grey values	–	Face detection
2008	[60]	●	Grey values	linear discriminant analysis	Face detection
2008	[108]	–	Curvature	–	Face detection
2008	[108]	●	Curvature + SIFT	*k*-nearest neighbor	Face detection
2007	[107]	●	Edges	Support Vector Machine (SVM)	Face detection
2006	[13]	●	Haar wavelet coefficients	Two SVMs	Face detection
2006	[76]	●	Haar wavelet coefficients	Cascaded AdaBoost	Face detection
2006	[18]	●	Haar-like + geometric	FloatBoost + MLP	Face detection
2006	[7]	●	Edges	PCA	Face detection
2005	[42]	●	Responses of Gabor filters	GMM + two SVMs	Face model
2004	[120]	–	Edges	–	Face detection
2004	[26]	●	Grey values	PRFR + AAM	Face detection
2002	[10]	●	Grey values	Recurrent neural networks	–
2001	[51]	●	Edges	Eye model + MLP	Face model
2011	our method	–	Image gradients	–	Face detection



(a) Image

(b) Image gradients

Figure 8.2.: Left: Example image of an eye and its surrounding area. Right: Vector field of image gradients for the eye image, where the intersection of the gradient vectors along the iris can be used as an estimation of the pupil's centre.

by analysing the vector field of image gradients (see Figure 8.2), which has been used for eye centre localisation previously. Kothari and Mitchell, for example, proposed a method that exploits the flow field character that arises due to the strong contrast between iris and sclera [59]. They use the orientation of each gradient vector to draw a line through the whole image and they increase an accumulator bin each time one such line passes through it. The accumulator bin where most of the lines intersect will be a maximum and will thus represent the estimated eye centre. They further propose to use a accumulator bin size of 5 pixels, which provides a reasonable trade-off between efficiency and effectiveness. However, they do not consider problems that arise due to eyebrows, eyelids, or glasses.

We have adopted the concept of analysing the vector field of image gradients and derived a mathematical formulation of the characteristics of the vector field (see Section 2.2.2), which can also be used for eye centre localisation by incorporating prior knowledge.

8.2.1. Incorporating Prior Knowledge

Under some conditions, the maximum of the objective function $J(\boldsymbol{c})$, see Equation 2.2, will not be well defined or there will also be local maxima that confuse the iterative scheme and lead to wrong centre estimates. For example, dominant eyelids and eyelashes or wrinkles in combination with a low contrast between iris and sclera can lead to wrong estimates. Therefore, we propose to incorporate prior knowledge about the eye appearance to increase robustness. Since the pupil is usually dark compared to sclera and skin, we apply a weight w_c for each possible centre $\boldsymbol{c}$ such that dark centres are more likely to be a correct centre estimate than bright centres. Integrating this into the objective function leads to:

$$J(\boldsymbol{c}) = \frac{1}{N} \sum_{i=1}^{N} w_c \left(\boldsymbol{d}_i^T \boldsymbol{g}_i \right)^2 \quad , \tag{8.1}$$

where $w_{\boldsymbol{c}} = I^*(c_x, c_y)$ is the grey value at (c_x, c_y) of the smoothed and inverted input image I^*. The image must be smoothed, e.g. by a Gaussian filter, to avoid problems that arise due to bright outliers such as reflections of glasses. The values of the modified objective function will not be very sensitive to changes in the parameters of the low-pass filter, we therefore suggest to use standard parameters relative to the ROI size.

Figure 8.3 demonstrates the improvement of incorporating prior knowledge into the objective function; since the gradients of the dominant eyelid and eyelashes contribute more to the sum of dot products than the gradients of iris and pupil, the maximum is not as significant as it is if prior knowledge is incorporated. The maximum not only becomes more distinctive, but also robustness improves significantly, if we apply this modification to the objective function. A major advantage of our gradient-based approach is that we can easily integrate the weight $w_{\boldsymbol{c}}$ into the existing iterative algorithm to efficiently obtain the maximum of Equation 8.1.

8.2.2. Postprocessing

The proposed summation of weighted squared dot products yields accurate results if the image contains the eye. However, when we apply the multi-stage scheme described in Figure 8.1, the rough eye regions sometimes also contain other structures such as hair, eyebrows, or glasses. Especially, hair and strong reflections in glasses show significant image gradients with different orientation than the image gradients of the pupil and the iris; hence the estimation of the eye centres may be wrong. We therefore propose a postprocessing step to overcome these problems; we apply a threshold on the objective function, based on the maximum value, and remove all remaining values that are connected to one of the borders as shown in Figure 8.4. Then, we determine the maximum of the remaining values and use its position as centre estimate. Based on our experiments the value of this threshold does not significantly influence the centre estimates, we suggest to set this threshold to 90% of the overall maximum.

8.3. Evaluation

We have chosen the BioID database for evaluation, since it is the most challenging set of images for eye centre localisation and many recent results are available. The database consists of 1521 grey level images of 23 different subjects and has been taken in different locations and at different daytimes; this results in unconstrained illumination comparable to outdoor scenes. In addition to the changes in illumination, the position

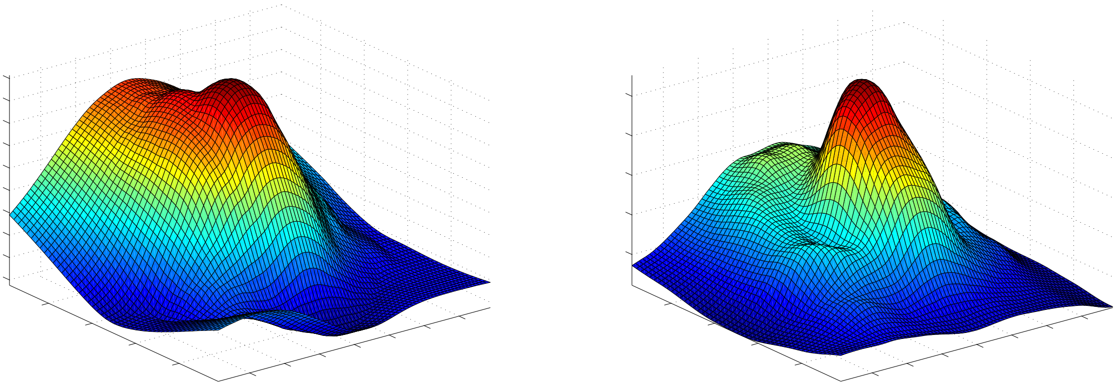


Figure 8.3.: The evaluation of Equation 8.1 for the eye image shown in Figure 8.2. Left: Without any prior knowledge, i.e. $\forall c \, : \, w_c = 1$, the global maximum is indistinct due to the dominant image gradients of the eyelid and the eyelashes. Right: Incorporating prior knowledge yields a more pronounced global maximum.

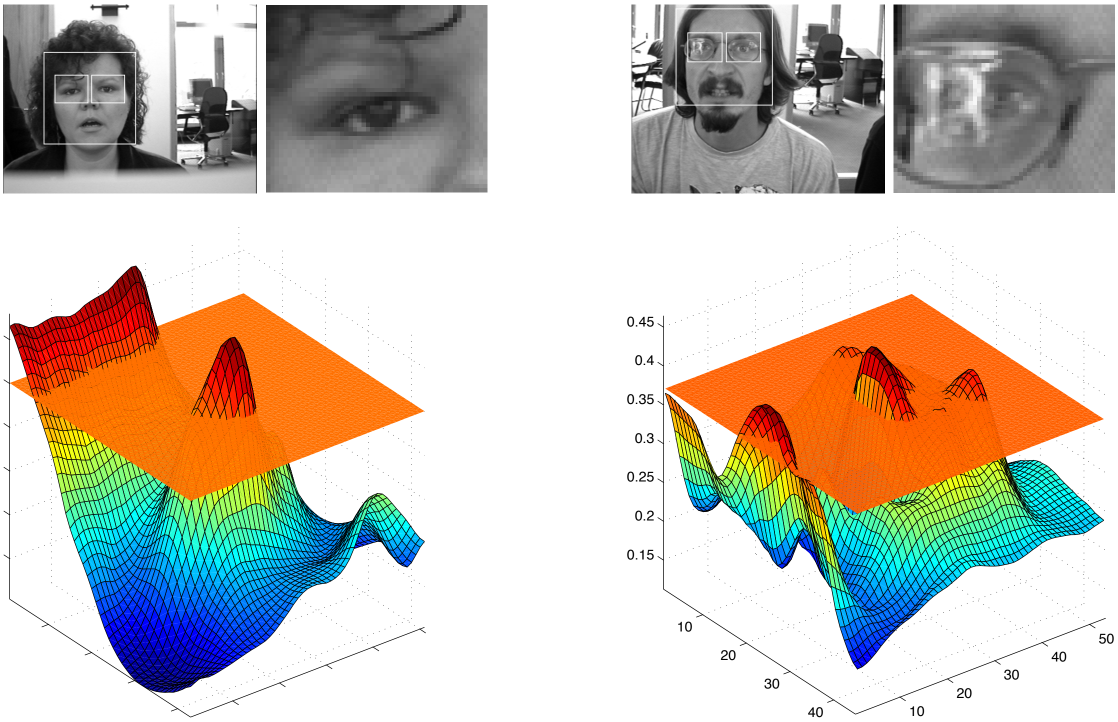


Figure 8.4.: Left panels: Example of a rough eye region, which contains strands of hair; the image gradients of these structures lead to a maximum of the objective function that is located at the border. If we apply a threshold (solid plane) and remove all values above the threshold that are near the border, the peak in the middle remains and corresponds to the correct eye's centre. Note that changing the threshold slightly does not have a strong influence. Right panels: The same holds for eye regions that contains glasses with strong reflections.

of the subjects change as well as their pose. Moreover, several subjects wear glasses and some subjects have curled hair near to the eye centres; in some images the eyes are closed and the head is turned away from the camera or strongly affected by shadows. In few images the eyes are even completely hidden by strong reflections on the glasses. Therefore, the BioID database is considered as one the most challenging database that reflects realistic conditions. The image quality and the image size (286×384) is approximately equal to the quality of a low-resolution (QVGA) webcam and the left and right eye centres are annotated and provided together with the images.

We perform the multi-stage scheme described in Figure 8.1, where the position of the face is detected first. Therefore, we apply a boosted cascade face detector that proved to be effective and accurate on several benchmarks [111]. Based on the position of the detected face, we extract rough eye regions relative to the size of the detected face. While anthropometric relations are often used to extract these regions, these relations do not generally hold for the area determined by the face detector. Here, we used 100 randomly selected images of the BioID database to evaluate the relative positions of the left and the right eye centres based on the detected face. Let (x, y) be the upper left corner and W, H the width and height of the detected face. Then, the mean of the right eye centre is located at $(x + 0.3 * W, y + 0.4 * H)$ and the mean of the left centre is at position $(x + 0.7 * W, y + 0.4 * H)$. Based on these estimated positions we extract rough regions for each eye with a size of $(0.35 * W) \times (0.3 * H)$—compare Figure 8.1 or 8.4. The rough eye regions are then used to estimate the eye centres accurately by applying the proposed approach.

As accuracy measure for the estimated eye centres, we evaluate the *normalised error*, which corresponds to the worst of both eye estimations; this measure was introduced by Jesorsky et al. and is defined as:

$$e \leq \frac{\max\left(e_{\text{left}}, e_{\text{right}}\right)}{d} \quad , \tag{8.2}$$

where e_{left}, e_{left} are the Euclidean distances between the estimated and the correct left and right eye centres, and d is the distance between the correct eye centres. When analysing the performance of an approach for eye localisation, this measure has the following characteristics: $e = 0.25$ is approximately the distance between the eye centre and the eye corners, $e = 0.10$ roughly corresponds to the diameter of the iris, and $e = 0.05$ equals the diameter of the pupil. Thus, an approach that should be used for eye tracking should not only provide high performance for $e \leq 0.25$, but must yield accurate results for $e \leq 0.05$. An error of slightly less than or equal to 0.25 will only indicate that the estimated centre are within the eye, but this estimation cannot be used to perform accurate eye tracking. We therefore focus on the performance that is obtained for $e \ll 0.25$, when we compare with state-of-the-art methods. Since in some published articles the normalised error is used in a non-standard way, we also provide the measures

$$e_{\text{min}} \leq \frac{\min\left(e_{\text{left}}, e_{\text{right}}\right)}{d} \quad \text{and} \quad e_{\text{avg}} \leq \frac{\left(e_{\text{left}} + e_{\text{right}}\right)}{2\,d} \;, \tag{8.3}$$

to give an upper bound as well as an averaged error.

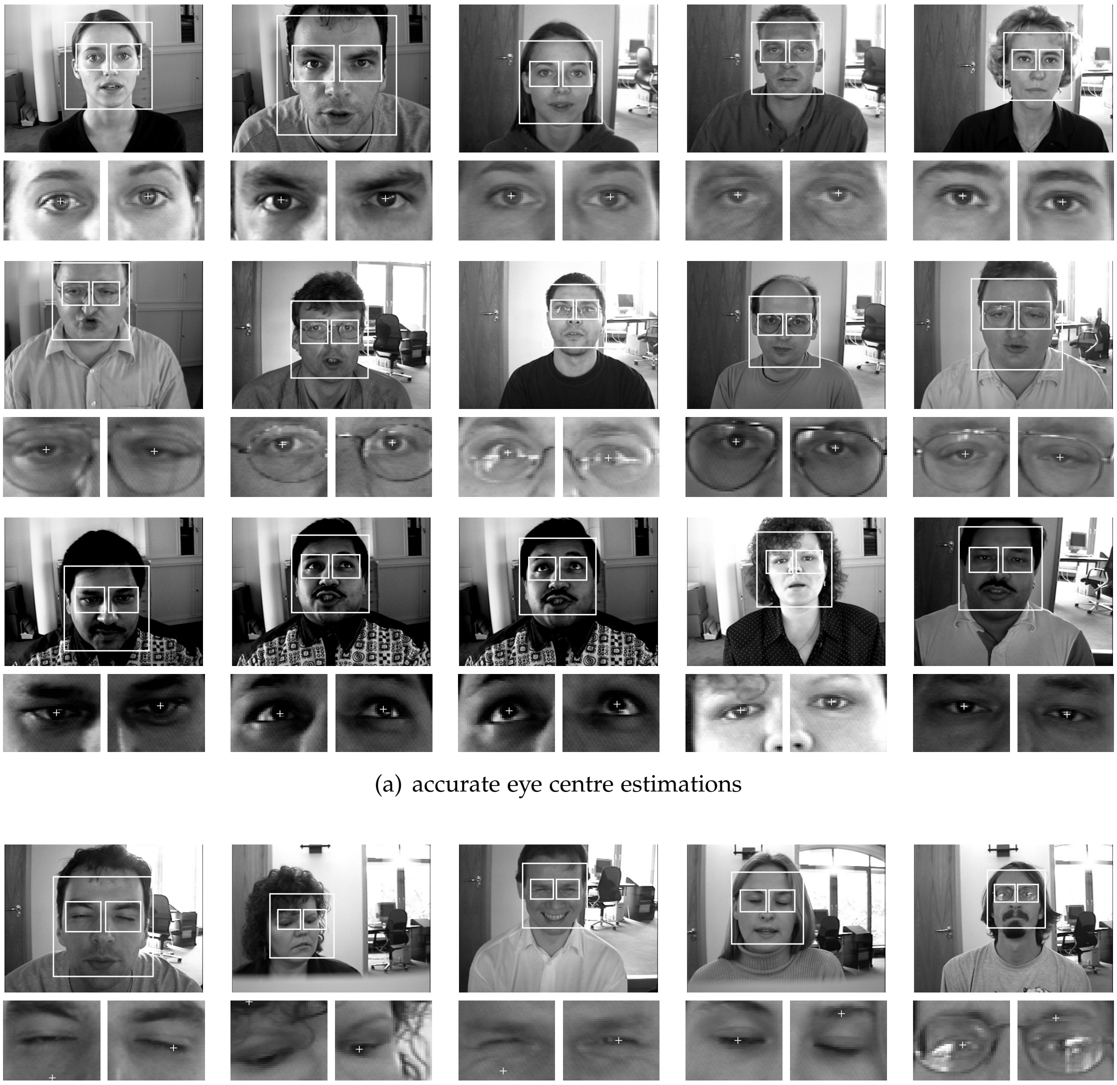
8.3.1. Results

Figure 8.5 shows the qualitative results of the proposed approach; we can observe that our approach yields accurate centre estimations not only for images containing dominant pupils (first row of Figure 8.5), but also in the presence of glasses (second row), shadows, low contrast, or strands of hair (third row). This demonstrates the robustness and proves that our approach can successfully deal with several, severe problems that arise in realistic scenarios. Our approach yields inaccurate estimations if the eyes are (almost) closed or strong reflections on the glasses occur (last row). In these cases, the gradient orientations of the pupil and the iris are affected by "noise" and hence their contribution to the sum of squared dot products is less than the contribution of the gradients around the eyebrow or eyelid. In some cases, the eyes are (almost) closed or hardly visible, which makes it difficult to estimate the eye centres even manually.

Figure 8.6 describes the quantitative results of the proposed method, where the accuracy measures e, e_{min}, and e_{avg} are illustrated. By using the standard definition of the normalised error, Equation 8.2, our approach yields an accuracy of 82.5% for pupil localisation ($e \leq 0.05$), which indicates that the estimates centres are located within the pupil with high probability; therefore, our approach can be used for eye tracking applications. Due to the fact that the BioID database contains some images with closed eyes, the performance will even increase if these images are left out. In the case of iris localisation ($e \leq 0.10$), the estimated centres lie within the iris with a probability of 93.4%, which will also further increase if images with closed eyes are left out. Only in rare cases (2.0%), the estimated centre is located outside a circle centred within the pupil and with a diameter equal to the distance between the eye corners— for example see Figure 8.5 (last row, first image).

8.3.2. Comparison With State of the Art

We extensively compare our method with state of the art methods that have been applied to the BioID images as well. Instead of elaborating these methods for eye centre localisation in detail, we refer to Table 8.1 for an overview. For comparison we evaluate the performance for different values of the normalised error e to obtain a characteristic curve, i.e. see Figure 8.6 "worse eye", which we will call *worse eye characteristic* (WEC). The WEC is roughly similar to the well-known receiver operator



(a) accurate eye centre estimations

(b) inaccurate eye centre estimations

Figure 8.5.: Sample images of accurate and inaccurate results for eye centre localisation on the BioID database. The estimated centres are depicted by white crosses. Note that the estimated centres may be difficult to identify due to low printer resolution.

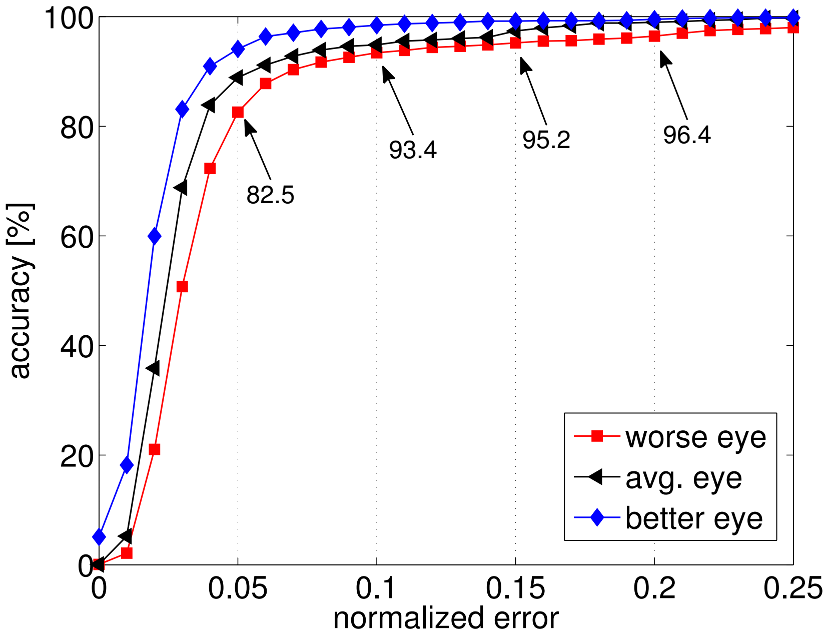


Figure 8.6.: Quantitative analysis of the proposed approach for the BioID database. In order to give upper and lower bounds, the accuracy versus the minimum (better eye, e_{min}), the maximum (worse eye, e) and the average (avg. eye, e_{avg}) normalised error are shown; some characteristic values are given explicitly.

characteristic (ROC) and can be analysed in several ways. As mentioned previously, it depends on the application, which e should be applied to compare different methods, e.g. for eye tracking applications a high performance for $e \leq 0.05$ is required, whereas for applications that use the overall eye position, such as face matching, comparing the performance for $e \leq 0.25$ will be more appropriate. In order to compare the overall performance, i.e. for different e, the area under the WEC can be used. However, the WEC of other methods is often not available. We therefore compare the methods for a discretised $e \in \{0.05, 0.10, 0.15, 0.20, 0.25\}$. Furthermore, we also evaluate the rank of each method according to the discretised e, which is roughly inversely proportional to the area under the WEC.

Table 8.2 compares our method with the state-of-the-art methods we have mentioned earlier in Table 8.1. If the performances for $e \in \{0.05, 0.10, 0.15, 0.20, 0.25\}$ were not provided by the authors explicitly, but a WEC is shown, we measured the values accurately from the WEC. Note that for some methods the authors evaluated the performance only for few e, see for example [18] or [120].

We can observe that for all e our method performs only 2% worse on average compared to the best method for each e. For example, the method proposed by Valenti and Gevers yields a performance of 84.1% for $e \leq 0.05$, whereas our method yields a performance of 82.5%. However, Valenti and Gevers reported that their method, which is based on isophotes in combination with a mean-shift clustering, SIFT features, and a k nearest neighbour for classification, will produce unstable centre estimations when it is applied to eye tracking with several images per second. Hence, our method can be considered as one of the best methods for accurate eye centre localisation. Furthermore, our method has significantly less computational complexity compared to that of Valenti and Gevers, since it requires neither clustering nor a classifier. Comparing those methods that do not involve any kind of learning scheme, our method achieves the best performance by far (82.5% for $e \leq 0.05$); second place: method by Valenti and Gevers (MIC) with 77.5%; third place: method by Asadifard and Shanbezadeh with 47%.

Our method achieves the second best performance (93.4%) in the case of iris location ($e \leq 0.10$); only the method by Cristinacce et al. yields a significant improvement (96.0%)—however, this improvement implies, again, a higher computational complexity compared to our method, which is solely based on simple dot products. For larger normalised errors, e.g. $e \leq 0.15$, $e \leq 0.20$, or $e \leq 0.25$, our method performs comparable to other methods.

Table 8.3 shows the corresponding ranks of the performances; we can clearly identify that there is no single method that performs superior for all values of e. Exemplarily, the method proposed by Türkan et al. achieves accurate estimations for detecting the overall eye locations, *i.e.* $e \leq 0.20$ and $e \leq 0.25$, but it fails for iris localisation ($e \leq 0.10$) and pupil localisation ($e \leq 0.05$) with a 13th place in both cases. Moreover, the method proposed by Cristinacce et al. ranks first for $e \leq 0.10$ and $e \leq 0.15$, but it ranks only 8th for $e \leq 0.05$. In contrast, our method ranks 2nd for both pupil and iris localisation and ranks 3rd and 4th for larger e. Hence, our method does not yield the best result for one single e, but if we evaluate the average rank, our method achieves the best result (3.0). Compared with the method that yields the second best average rank (3.4, Valenti and Gevers, MIC+SIFT+kNN) our method shows significantly less variance according to the individual ranks.

In total, our method performs comparable to state-of-the-art-methods when looking for a particular e, but it yields the best average performance over all values of e. Hence, our method proves to be powerful for several problems such as eye centre localisation ($e \leq 0.05$), iris localisation ($e \leq 0.10$), and eye localisation ($e \leq 0.25$). Comparing only those methods that do not apply any learning scheme, our method achieves significant improvements for the more difficult tasks, i.e. 5% improvement for $e \leq 0.05$, 7% for $e \leq 0.10$, and 2.6% for $e \leq 0.15$.

Table 8.2.: Comparison of the performance for eye detection on the BioID database. Brackets indicate values that have been accurately measured from author's graphs. (∗) Images with closed eyes and glasses were omitted. (•) Methods that do not involve any kind of learning or model scheme. Since some authors did not provide any graphical evaluation of the performance, e.g. by using a WEC curve, intermediate values could not be estimated—these missing values are denoted by "–".

Method	$e \leq 0.05$	$e \leq 0.10$	$e \leq 0.15$	$e \leq 0.20$	$e \leq 0.25$	Remarks
[6]	47.0%	86.0%	89.0%	93.0%	96.0%	(∗), (•)
[60]	65.0%	87.0%	–	–	98.8%	
[108]	77.2%	82.1%	(86.2%)	(93.8%)	96.4%	MIC, (•)
[108]	**84.1%**	90.9%	(93.8%)	(97.0%)	98.5%	MIC+SIFT+*k*NN
[107]	(18.6%)	73.7%	(94.2%)	**(98.7%)**	**99.6%**	
[13]	62.0%	85.2%	87.6%	91.6%	96.1%	
[76]	(75.0%)	93.0%	(95.8%)	(96.4%)	(97.0%)	
[18]	–	89.7%	–	–	95.7%	
[7]	(44.0%)	81.7%	(92.6%)	(96.0%)	97.4%	(•)
[42]	(58.6%)	(75.0%)	(80.8%)	(87.6%)	(91.0%)	
[120]	–	–	–	–	94.8%	(•)
[26]	(57.0%)	**96.0%**	**(96.5%)**	(97.0%)	(97.1%)	
[10]	(37.0%)	(86.0%)	(95.0%)	(97.5%)	(98.0%)	
[51]	(38.0%)	(78.8%)	(84.7%)	(87.2%)	91.8%	
our method	**82.5%**	**93.4%**	**95.2%**	**96.4%**	**98.0%**	(•)

8.4. Discussion

We have demonstrated that the problem of eye centre localisation can be solved efficiently and with high accuracy by using our novel approach that is based on image gradients. For every pixel, we compute the squared dot product between the displacement vector of a centre candidate and the image gradient. By summing up these squared dot products we derive a simple objective function that needs to be maximised. The position of the maximum then corresponds to the position where most image gradients (if extended in both directions) intersect. Furthermore, we have demonstrated that prior knowledge such as grey level intensities can easily be integrated into the objective function to increase robustness.

Since only image gradients and simple mathematical operations are involved, our method yields low computational complexity. Moreover, our method is invariant to

Table 8.3.: Comparison of the ranks according to the performance of Table 8.2.

Method	$e \leq 0.05$	$e \leq 0.10$	$e \leq 0.15$	$e \leq 0.20$	$e \leq 0.25$	avg. rank
[6]	9	7	8	7	10	8.2
[60]	5	6	–	–	2	4.3
[108]	3	9	10	6	8	7.2
[108]	**1**	4	6	3	3	**3.4**
[107]	13	13	5	**1**	**1**	6.6
[13]	6	8	9	8	9	8.0
[76]	4	3	2	4	7	4.0
[18]	–	5	–	–	11	8.0
[7]	10	10	7	5	5	7.4
[42]	7	12	12	9	14	10.8
[120]	–	–	–	–	12	12.0
[26]	8	**1**	**1**	3	6	3.8
[10]	12	7	4	2	4	5.8
[51]	11	11	11	10	13	11.2
our method	**2**	**2**	**3**	**4**	**4**	**3.0**

rotation and linear changes in illumination. Due to these properties, our method can be applied to several (real-time) applications that require a high accuracy such as eye tracking, industrial inspection of circular objects, or medical imaging analysis (cell tracking).

We extensively evaluated our method on one of the most challenging databases, which demonstrates the robustness to changes in illumination, pose, scale, and occlusion—even for low-resolution images. Compared to several state-of-the-art methods, our method ranks second for special scenarios such as pupil localisation or iris localisation, and it ranks first if the average performance over several scenarios, e.g. pupil localisation, iris localisation, and overall eye localisation, is evaluated. Compared to methods that do not involve any kind of learning, such as a support vector machine or a *k* nearest neighbour classifier, our method achieves a significant improvement of 5–7%. Moreover, we have created a short video sequence that demonstrates the robustness and accuracy of our novel approach, see `http://www.youtube.com/watch?v=aGmGyFLQAFM`.

We believe that our method can be further improved by incorporating context and more facial features (eye centre must be located between the eye corners). Furthermore, a learning method used to reduce the number of wrong centre estimates may also improve the results significantly.

9. Summary and Outlook

I have presented several methods for image preprocessing, feature extraction, and novelty detection, and I have demonstrated that these methods yield accurate and competitive results for benchmark datasets as well as in real-world applications. Moreover, I have already discussed the results individually at the end of the corresponding chapters. Now, I will only pick up few points, try to take a broader view and give an outlook on possible requirements of future machine vision systems.

Machine vision systems for inspection and novelty detection have mainly been designed and tuned for a particular application. It is, of course, very important to bring research and industry permanently closer, but I believe that machine vision systems should be evaluated more systematically and put into a general framework with standard requirements, for instance the ones we have introduced in Chapter 7. In other areas such as face detection or eye tracking, various public benchmark datasets have been proposed to make fair comparisons and many researchers have tried to develop methods that perform well on all of these datasets. This standardisation will push research forward and will bring research faster into real-life—nowadays, online face detection is already integrated in state-of-the-art consumer digital cameras and low-cost eye tracking devices are used for marketing analysis, for example. I believe that this progress can be achieved as well for industrial applications such as surface inspection or inspection of semi-conductor components once industrial datasets become public; the dataset for weakly supervised learning of texture defects provided by the company Robert Bosch GmbH can be seen as a first step in this direction.

I have demonstrated in a series of experiments that there is no novelty-detection method that yields superior performance on various datasets; in most cases performance is roughly comparable. In the field of machine learning, where novelty-detection methods originated, several techniques, such as the usage of multiple kernels instead of a single kernel or cascades of support vector machines, have been proposed to further improve performance. Even though these optimisations may lead to more powerful learning methods, I think that in case of a machine vision system the input features of the learning scheme are more important than the learning scheme itself. For example, our simple gradient-based approach for eye centre localisation, see Chapter 8, does not employ any kind of learning and it outperforms approaches that use grey values as input for multiple layers of support vector machines. I do not mean that we should not derive new learning algorithms, but in case of a machine vision system preprocessing and the extraction of appropriate image features are more crucial.

There is a trend in machine vision, and especially in machine learning, that novel methods are accepted most easily if they contain a sophisticated mathematical derivation several pages long; unfortunately, this compels researchers to take an existing, complex method and to modify it only slightly. As a result, methods become more and more complex and are no longer applicable by practitioners. In this thesis, I have described methods that are easy to implement—even for practitioners.

Besides simplicity, I believe that future machine vision systems must be more efficient computationally to be applied in non-industrial applications, for which energy consumption plays a crucial role. In automotive applications, for example, various driver assistance systems have been developed such as parking assistance or obstacle detection. Nowadays, the energy that is required for these assistance systems does not significantly affect driving, since the increased fuel consumption is barely noticed. However, if all these systems are integrated into electric cars, we would have to review every assistance system, not only the vision-based, to further increase efficiency. In the same context, we have to think of new ways for evaluating machine vision systems, since the computation time does not always correspond to energy consumption. For example, a master machine vision system could decide to (partially) switch on and off front and rear camera based on the current driving—in a traffic jam, for instance, assistance systems such as obstacle detection or pedestrian detection are not required.

Overall, even today's industrial products are already highly complex and cannot be inspected manually with high accuracy and reliability. Therefore, efficient but simple machine vision systems for industrial inspection are becoming more and more important.

A. Solving an Eigenvalue Problem in Feature Space

For computing principal components in some feature space we have to find non-negative Eigenvalues λ and non-zero Eigenvectors $\boldsymbol{v}$ satisfying

$$\lambda\, \boldsymbol{v} = \boldsymbol{C}\, \boldsymbol{v} \ , \tag{A.1}$$

where $\boldsymbol{C}$ is the covariance matrix of the samples $\boldsymbol{x}_i$ in feature space:

$$\boldsymbol{C} = \frac{1}{n} \sum_{i=1}^{n} \hat{\phi}(\boldsymbol{x}_i) \hat{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} \tag{A.2}$$

and

$$\hat{\phi}(\boldsymbol{x}_i) = \phi(\boldsymbol{x}_i) - \phi_0 = \phi(\boldsymbol{x}_i) - \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x}_i) \ . \tag{A.3}$$

Since all possible solutions $\boldsymbol{v}$ lie in the span of $\hat{\phi}(\boldsymbol{x}_1), \ldots, \hat{\phi}(\boldsymbol{x}_n)$, we can also solve the equivalent system

$$\forall l = 1, ..., n : \ \lambda(\hat{\phi}(\boldsymbol{x}_l)^{\mathrm{T}} \boldsymbol{v}) = \hat{\phi}(\boldsymbol{x}_l)^{\mathrm{T}} \boldsymbol{C} \boldsymbol{v} \ , \tag{A.4}$$

and there exist coefficients $\alpha_1, \ldots, \alpha_n$ such that the Eigenvectors can be written as a linear combination of the data samples in feature space:

$$\boldsymbol{v} = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \ . \tag{A.5}$$

By substituting (A.2) and (A.5) into (A.4) and rearraging we have to solve the Eigenvalue problem

$$n \lambda \boldsymbol{\alpha} = \boldsymbol{K} \boldsymbol{\alpha} \ , \tag{A.6}$$

where $\boldsymbol{K}_{ij} := K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \hat{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} \hat{\phi}(\boldsymbol{x}_j)$ denotes the kernel matrix. We ensure that the Eigenvectors $\boldsymbol{v}$ in feature space have unit length by

$$\begin{aligned} \|\boldsymbol{v}\|_2^2 = \boldsymbol{v}^{\mathrm{T}} \boldsymbol{v} &= \sum_{i,j=1}^{n} \alpha_i \alpha_j \hat{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} \hat{\phi}(\boldsymbol{x}_j) \\ &= \boldsymbol{\alpha}^{\mathrm{T}} \boldsymbol{K} \boldsymbol{\alpha} = n\, \lambda\, \boldsymbol{\alpha}^{\mathrm{T}} \boldsymbol{\alpha} = 1 \end{aligned} \tag{A.7}$$

and therefore we scale $\boldsymbol{\alpha}$ such that

$$\|\boldsymbol{\alpha}\|_2 = \frac{1}{\sqrt{n\lambda}} \quad . \tag{A.8}$$

Since we don't have the vectors $\phi(\boldsymbol{x}_i)$ explicitly, we cannot compute the mean $\phi_0 = \frac{1}{n}\sum_{l=1}^{n}\phi(\boldsymbol{x}_l)$ in $\mathcal{H}$. Instead, we adapt the kernel matrix such that only dot products of centered data points appear:

$$\begin{aligned}\hat{\boldsymbol{K}}_{ij} &= (\phi(\boldsymbol{x}_i) - \phi_0)^{\mathrm{T}} \left(\phi(\boldsymbol{x}_j) - \phi_0\right) \\ &= \boldsymbol{K}_{ij} - \frac{1}{n}\sum_{l=1}^{n}\boldsymbol{K}_{il} - \frac{1}{n}\sum_{k=1}^{n}\boldsymbol{K}_{kj} + \mu \quad ,\end{aligned} \tag{A.9}$$

where

$$\mu = \frac{1}{n^2}\sum_{h,m=1}^{n}\boldsymbol{K}_{hm} \quad . \tag{A.10}$$

Thus, the kernel matrix $\boldsymbol{K}$ in (A.6) can be substituted by

$$\hat{\boldsymbol{K}} = \boldsymbol{K} - \mathbf{1}_n\boldsymbol{K} - \boldsymbol{K}\mathbf{1}_n + \mathbf{1}_n\boldsymbol{K}\mathbf{1}_n \quad , \tag{A.11}$$

where $(\mathbf{1}_n)_{ij} = 1/n$.

Bibliography

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing. *RCA Engineer*, 29(6):33–41, 1984.

[2] Yael Adini, Yael Moses, and Shimon Ullman. Face recognition: the problem of compensating for changes in illumination direction. *IEEE TPAMI*, 19:721–732, 1997.

[3] Realtime Technology AG. *RTT DeltaTex – Virtual material design in realtime*, 2010. Software available at `http://www.realtime-technology.com`.

[4] Manoj Aggarwal, Hong Hua, and Narendra Ahuja. On cosine-fourth and vignetting effects in real lenses. In *ICCV*, pages 472–479, 2001.

[5] Erling D. Andersen, Bo Jensen, Jens Jensen, Rune Sandvik, and Ulf Worsoe. Mosek version 6. Technical Report TR-2009-3, MOSEK ApS, Copenhagen, Denmark, 2009.

[6] Mansour Asadifard and Jamshid Shanbezadeh. Automatic adaptive center of pupil detection using face detection and cdf analysis. In S. I. Ao, Oscar Castillo, Craig Douglas, David Dagan Feng, and Jeong-A Lee, editors, *Proceedings of the MultiConference of Engineers and Computer Scientists*, volume I of *Lecture Notes in Computer Science*, pages 130–133, Hong Kong, 2010. International Association of Engineers, Newswood Limited.

[7] S. Asteriadis, S. Asteriadis, N. Nikolaidis, A. Hajdu, and I." Pitas. An eye detection algorithm using pixel to edge information. In *Proceedings of the 2nd International Symposium on Control, Communications, and Signal Processing*, Marrakech, Morocco, 2006. EURASIP.

[8] T. J. Atherton and Darren J. Kerbyson. Size invariant circle detection. *Image Vision Comp.*, 17(11):795–803, 1999.

[9] I.B. Ayed, N. Hennane, and A. Mitiche. Unsupervised variational image segmentation/classification using a Weibull observation model. *IEEE Transactions on Image Processing*, 15(11):3431–3439, 2006.

[10] S. Behnke. Learning face localization using hierarchical recurrent networks. In *Proceedings of the International Conference on Artificial Neural Networks*, Lecture Notes in Computer Science, pages 135–135. Springer, 2002.

[11] Martin Böhme, Andre Meyer, Thomas Martinetz, and Erhardt Barth. Remote eye tracking: State of the art and directions for future development. In *The 2nd Conference on Communication by Gaze Interaction*, pages 10–15, Italy, 2006.

[12] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In David Haussler, editor, *Proc. of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

[13] Paola Campadelli, Raffaella Lanzarotti, and Giuseppe Lipori. Precise eye localization through a general-to-specific model definition. In *Proceedings of the 17th British Machine Vision Conference*, volume I, pages 187–196, Edingburgh, England, 2006.

[14] Michele Ceccarelli, Alfredo Petrosino, and Giuliano Laccetti. Circle detection based on orientation matching. In *ICIAP*, pages 119–124. IEEE Computer Society, 2001.

[15] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[16] Chuan-Yu Chang, Chun-Hsi Li, Jia-Wei Chang, and MuDer Jeng. An unsupervised neural network approach for automatic semiconductor wafer defect inspection. *Expert Syst. Appl*, 36(1):950–958, 2009.

[17] Chuan Yu Chang, Chun Hsi Li, Si Yan Lin, and MuDer Jeng. Application of two hopfield neural networks for automatic four-element LED inspection. *IEEE Trans. Systems, Man and Cybernetics*, 39(3):352–365, 2009.

[18] D. Chen, X. Tang, Z. Ou, and N. Xi. A hierarchical floatboost and mlp classifier for mobile phone embedded eye location system. In *Proceedings of the 3rd International Symposium on Neural Networks*, Lecture Notes in Computer Science, pages 20–25, Chengdu, China, 2006. Springer.

[19] Wen-Chin Chen and Shou-Wen Hsu. A neural-network approach for an automatic LED inspection system. *Expert Systems with Applications*, 33(2):531–537, 2007.

[20] Y. Q. Chen, M. S. Nixon, and D. W. Thomas. Statistical geometrical features for texture classification. *Pattern Recognition*, 28(4):537–552, 1995.

[21] S. Chiu and M. Perng. Reflection-area-based feature descriptor for solder joint inspection. *Machine Vision and Applications*, 18(2):95–106, 2007.

[22] P. B. Chou, A. R. Rao, M. C. Sturzenbecker, F. Y. Wu, and V. H. Brecher. Automatic defect classification for semiconductor manufacturing. *Machine Vision and Applications*, 9(4):201–214, 1997.

[23] M. Connolly and D. Van Essen. The representation of the visual field in parvicellular and magnocellular layers of the lateral geniculate nucleus in the macaque monkey. *The Journal of comparative neurology*, 226(4):544–564, 1984.

[24] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[25] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge, U.K., 2000.

[26] D. Cristinacce, T. Cootes, and I. Scott. A multi-stage approach to facial feature detection. In *Proceedings of the 15th British Machine Vision Conference*, pages 277–286, London, England, 2004.

[27] K.J. Dana, B. Van Ginneken, S.K. Nayar, and J.J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics (TOG)*, 18(1):1–34, 1999.

[28] M. Driels and C. Lee. Feature selection for automatic visual inspection of solder joints. *The Int. Journal of Advanced Manufacturing Technology*, 3:3–32, 1988.

[29] R. O. Duda and P. E. Hart. Use of the Hough transform to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[30] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis*. John Wiley & Sons, New York, NY, 1973.

[31] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240, 1967.

[32] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, pages 341–346, New York, NY, USA, 2001. ACM.

[33] M. H. Fadzil and C. J. Weng. LED cosmetic flaw inspection system. *Pattern Analysis and Applications*, 1(1):62–70, 1998.

[34] R.E. Fan, P.H. Chen, and C.J. Lin. Working set selection using second order information for training support vector machines. *The Journal of Machine Learning Research*, 6:1889–1918, 2005.

[35] CNS Ganesh Murthy and YV Venkatesh. Encoded pattern classification using constructive learning algorithms based on learning vector quantization. *Neural Networks, Elsevier*, 11(2):315–322, 1998.

[36] J.M. Geusebroek and A.W.M. Smeulders. Fragmentation in the vision of scenes. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 130–135. IEEE Computer Society, 2003.

[37] J.M. Geusebroek and A.W.M. Smeulders. A six-stimulus theory for stochastic texture. *International Journal of Computer Vision*, 62(1):7–16, 2005.

[38] J.H. Gove. Moment and maximum likelihood estimators for Weibull distributions under length-and area-biased sampling. *Environmental and Ecological Statistics*, 10(4):455–467, 2003.

[39] G. H. Granlund. Fourier preprocessing for hand print character recognition. *IEEE Trans. Computer*, 21(2):195–201, 1972.

[40] Sheng Uei Guan, Pin Xie, and Hong Li. A golden-block-based self-refining scheme for repetitive patterned wafer inspections. *Machine Vision and Applications*, 13(5):314–321, 2003.

[41] Roland Haitz, Fred Kish, Jeff Tsao, and Jeff Nelson. The case for a national research program on semiconductor lighting, April 2000.

[42] M. Hamouz, J. Kittler, JK Kamarainen, P. Paalanen, H. Kalviainen, and J. Matas. Feature-based affine-invariant localization of faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1490, 2005.

[43] J A Hanley and B J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, Apr 1982.

[44] T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Verlag, 2009.

[45] B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. Technical Report 1687, MIT Computer Science and Artificial Intelligence Lab, 2000.

[46] Heiko Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007.

[47] P. V. C. Hough. Methods and means to recognize complex patterns. U.S. Patent 3,069,654, 1962.

[48] John Illingworth and Josef Kittler. A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing*, 44(1):87–116, 1988.

[49] D. Ioannou, W. Huda, and A. F. Laine. Circle recognition through a 2D Hough transform and radius histogramming. *Image and Vision Computing*, 17(1):15–26, 1999.

[50] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall Information and Systems Science Series, Upper Saddle River, NJ, USA, 1989.

[51] O. Jesorsky, K. Kirchberg, and R. Frischholz. Robust face detection using the Hausdorff distance. In *Proceedings of the 3rd International Conference on Audio- and Video-Based Biometric Person Authentication*, Lecture Notes in Computer Science, pages 90–95, Halmstad, Sweden, 2001. Springer.

[52] H. Kauppinen, T. Seppanen, and M. Pietikainen. An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification. *IEEE TPAMI*, 17(2):201–207, 1995.

[53] P. Kierkegaard. A method for detection of circular arcs based on the Hough transform. *Machine Vision and Applications*, 5:249–263, 1992.

[54] J. Kim and H. Cho. Neural network-based inspection of solder joints using a circular illumination. *Image and Vision Computing*, 13(6):479–490, 1995.

[55] Seon Joo Kim and Marc Pollefeys. Robust radiometric calibration and vignetting correction. *IEEE TPAMI*, 30(4):562–576, 2008.

[56] C. Kimme, D. H. Ballard, and J. Sklansky. Finding circles by an array of accumulators. *Communications of the ACM*, 18(2):120–122, 1975.

[57] Sascha Klement, Fabian Timm, and Erhardt Barth. Illumination correction for image stitching. In *Proceedings of the International Conference on Imaging Theory and Applications (IMAGAPP)*, volume 1, pages 81–86, Algarve, Portugal, 2011. INSTICC.

[58] K. Ko and H. Cho. Solder joints inspection using a neural network and fuzzy rule-based classification method. *IEEE Transactions on Electronics Packaging Manufacturing*, 23(2):93–103, 2000.

[59] R. Kothari and J.L. Mitchell. Detection of eye locations in unconstrained visual images. In *Proceedings of the International Conference on Image Processing*, volume 3, pages 519–522. IEEE, IEEE Computer Society Press, 1996.

[60] B. Kroon, A. Hanjalic, and S.M.P. Maas. Eye localization for face matching: is it always useful and under what conditions? In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, pages 379–388, Ontario, Canada, 2008. ACM.

[61] A. Kumar. Computer-vision-based fabric defect detection: A survey. *IEEE Transactions on Industrial Electronics*, 55(1):348–363, 2008.

[62] A. Kumar and G.K.H. Pang. Defect detection in textured materials using Gabor filters. *Industry Applications, IEEE Transactions on*, 38(2):425–440, 2002.

[63] I. Kunttu, L. Lepisto, J. Rauhamaa, and A. Visa. Multiscale Fourier descriptors for defect image retrieval. *Pattern Recognition Letters*, 27(2):123–132, 2006.

[64] Kai Labusch, Fabian Timm, and Thomas Martinetz. Simple incremental one-class support vector classification. In Gerhard Rigoll, editor, *Proceedings of the 30th DAGM-Symposium on Pattern Recognition*, volume 5096 of *Lecture Notes in Computer Science*, pages 21–30. Springer, 2008.

[65] Anat Levin, Assaf Zomet, Shmuel Peleg, and Yair Weiss. Seamless image stitching in the gradient domain. In *ECCV*, pages 377–389. Springer-Verlag, 2004.

[66] M. D. Levine, Maulin R. Gandhi, and Jisnu Bhattacharyya. Image normalization for illumination compensation in facial images. unpublished, August 2004.

[67] Jeroen Lichtenauer, Emile A. Hendriks, and Marcel J. T. Reinders. Isophote properties as features for object detection. In *CVPR*, pages 649–654. IEEE Computer Society, 2005.

[68] Boštjan Likar, Max A. Viergever, and Franjo Pernuš. Retrospective Correction of MR Intensity Inhomogeneity by Information Minimization. *IEEE Transactions on Medical Imaging*, 20(12):1398–1410, December 2001.

[69] Hong-Dar Lin. Automated defect inspection of light-emitting diode chips using neural network and statistical approaches. *Expert Systems with Applications*, 36(1):219–226, 2009.

[70] Hong-Dar Lin and Chung-Yu Chung. A wavelet-based neural network applied to surface defect detection of LED chips. In Derong Liu, Shumin Fei, Zeng-Guang Hou, Huaguang Zhang, and Changyin Sun, editors, *Proceedings of the 4th International Symposium on Neural Networks*, volume 4492 of *Lecture Notes in Computer Science*, pages 785–792. Springer, 2007.

[71] Y.H. Liu, Y.C. Liu, and Y.J. Chen. Fast support vector data descriptions for novelty detection. *IEEE Transactions on Neural Networks*, 21(8):1296–1313, 2010.

[72] Markos Markou and Sameer Singh. Novelty detection: A review - part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[73] Markos Markou and Sameer Singh. Novelty detection: A review - part 2: neural network based approaches. *Signal Processing*, 83(12):2499–2521, 2003.

[74] Babu M. Mehtre, Mohan S. Kankanhalli, and Wing Foon Lee. Shape measures for content based image retrieval: A comparison. *Inf. Process. Manage*, 33(3):319–337, 1997.

[75] L. G. Minor and J. Sklansky. The detection and segmentation of blobs in infrared images. *IEEE Trans. Systems, Man and Cybernetics*, 11:194–201, 1981.

[76] Z. Niu, S. Shan, S. Yan, X. Chen, and W. Gao. 2d cascaded adaboost for eye localization. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 2, pages 1216–1219, Hong Kong, 2006. IEEE Computer Society, IEEE Computer Society Press.

[77] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999.

[78] J. M. Ogden, E. H. Adelson, J. R. Bergen, and P. J. Burt. Pyramid-based computer graphics. *RCA Engineer*, pages 4–15, September/October 1985.

[79] T. Ong, Z. Samad, and M. Ratnam. Solder joint inspection with multi-angle imaging and an artificial neural network. *The Int. Journal of Advanced Manufacturing Technology*, 38(5–6):455–462, 2008.

[80] Emanuel Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[81] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

[82] W. Poechmueller, M. Glesner, L. Listl, and P. Mengel. Automatic classification of solder joint images. In *Proc. of the Int. Joint Conf. on Neural Networks*, volume 2, pages 933–940. IEEE Computer Society Press, 1991.

[83] T. Randen and J. H. Husoy. Filtering for texture classification: A comparative study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):291–310, 1999.

[84] Gunnar Rätsch. *Robust boosting via convex optimization: Theory and applications*. PhD thesis, Potsdam University, 2001.

[85] C.C. Reyes-Aldasoro. Retrospective shading correction algorithm based on signal envelope estimation. *Electronics Letters*, 45(9):454–456, 23 2009.

[86] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organisation in the brain. *Psychological Review*, 65(6):386–408, 1958.

[87] J. C. Russ. *The Image Processing Handbook*. CRC Press, 2007.

[88] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[89] B. Schölkopf, A. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[90] Bernhard Schölkopf and Alex Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[91] H.S. Scholte, S. Ghebreab, L. Waldorp, A.W.M. Smeulders, and V.A.F. Lamme. Brain responses strongly correlate with Weibull image statistics when processing natural images. *Journal of Vision*, 9(4):1–15, 2009.

[92] T. Serre, M. Kouh, C. Cadieu, U. Knoblich, G. Kreiman, and T. Poggio. A theory of object recognition: computations and circuits in the feedforward path of the ventral stream in primate visual cortex, cbcl mit paper (november 2005). Technical report, MIT, 2005.

[93] M Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):111–147, 1974.

[94] K. K. Sung. Learning and example selection for object and pattern detection. Technical Report 1572, MIT Computer Science and Artificial Intelligence Lab, 1996.

[95] R.E. Tarjan. Finding optimum branchings. *Networks*, 7(1):25–35, 1977.

[96] David M. J. Tax and Robert P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.

[97] David M. J. Tax and Robert P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[98] Fabian Timm and Erhardt Barth. Novelty detection for the inspection of light-emitting diodes. *Expert Systems with Applications*, 2010. under review.

[99] Fabian Timm and Erhardt Barth. Accurate eye centre localisation by means of gradients. In *Proceedings of the International Conference on Computer Theory and Applications (VISAPP)*, volume 1, pages 125–130, Algarve, Portugal, 2011. INSTICC.

[100] Fabian Timm and Erhardt Barth. Accurate, fast, and robust centre localisation for images of semiconductor components. In *Image Processing: Machine Vision Applications IV*, Electronic Imaging, San Francisco, USA, 2011. IS&T/SPIE.

[101] Fabian Timm and Erhardt Barth. Non-parametric texture defect detection using Weibull features. In *Image Processing: Machine Vision Applications IV*, Electronic Imaging, San Francisco, USA, 2011. IS&T/SPIE.

[102] Fabian Timm, Sascha Klement, and Thomas Martinetz. Fast model selection for MaxMinOver-based training of support vector machines. In *Proceedings of the 19th IEEE International Conference on Pattern Recognition (ICPR)*, pages 1–4. IEEE Computer Society, IEEE Computer Society Press, 2008.

[103] Fabian Timm, Sascha Klement, Thomas Martinetz, and Erhardt Barth. Welding inspection using novel specularity features and a one-class SVM. In *Proceedings of the Int. Conference on Computer Theory and Applications (VISAPP)*, volume 1, pages 146–153, Lisboa, Portugal, 2009. INSTICC.

[104] Fabian Timm and Thomas Martinetz. Statistical Fourier descriptors for defect image classification. In *Proceedings of the 20th International Conference on Pattern Recognition (ICPR)*, pages 4190–4193, Istanbul, Turkey, 2010. IEEE Computer Society Press.

[105] Fabian Timm, Thomas Martinetz, and Erhardt Barth. Optical inspection of welding seams. In *Computer Vision, Imaging and Computer Graphics: Theory and Applications, Revised Selected Papers*, volume 68 of *Communications in Computer Science and Information Science*, pages 269–282. Springer, 2010.

[106] Mihran Tuceryan and Anil K. Jain. Texture analysis. In C. H. Chen, L. F. Pau, and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, chapter 2, pages 235–276. World Scientific, Singapore, 1993.

[107] Mehmet Türkan, Montse Pardàs, and A. Enis Çetin. Human eye localization using edge projections. In Alpesh Ranchordas, Helder Araújo, and Jordi Vitrià, editors, *Proceedings of the 2nd International Conference on Computer Vision Theory and Applications*, pages 410–415, Barcelona, Spain, 2007. INSTICC, INSTICC.

[108] R. Valenti and T. Gevers. Accurate eye center location and tracking using isophote curvature. In *Proceedings of the International Conference on Computer Vision and*

Pattern Recognition, pages 1–8, Achorage, Alaska, 2008. IEEE Computer Society Press.

[109] Peter J. van Otterloo. *A Contour-oriented Approach to Shape Analysis*. Prentice Hall International (UK) Ltd., Hertfordshire, UK, 1991.

[110] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, Heidelberg, DE, 1995.

[111] P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[112] R. Walker and P. T. Jackway. Statistical geometric features: Extensions for cytological texture analysis. In *Proc. of the 13th Int. Conf. on Pattern Recognition*, pages 790–794. IEEE Computer Society Press, 1996.

[113] M. J. Wang and C. L. Huang. Evaluating the eye fatigue problem in wafer inspection. *IEEE Transactions on Semiconductor Manufacturing*, 3(17):444–447, 2004.

[114] M. Wu and J. Ye. A small sphere and large margin approach for novelty detection using training data with outliers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):2088–2092, 2009.

[115] X. Xie. A review of recent advances in surface defect detection using texture analysis techniques. *Electronic Letters on Computer Vision and Image Analysis*, 7(3):1–22, 2008.

[116] V. Yanulevskaya and JM Geusebroek. Significance of the Weibull distribution and its sub-models in natural image statistics. In *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications (VISAPP)*, volume 1, pages 355–362. INSTICC Press, 2009.

[117] H. K. Yuen, J. P. Princen, J. Illingworth, and J. V. Kittler. Comparative study of Hough transform methods for circle finding. *Image and Vision Comp.*, 8(1):71–77, 1990.

[118] D. S. Zhang and G. J. Lu. A comparative study of curvature scale space and Fourier descriptors for shape-based image retrieval. *Journal of Visual Communication and Image Representation*, 14(1):39–57, 2002.

[119] Y.J. Zheng, S. Lin, C. Kambhamettu, J.Y. Yu, and S.B. Kang. Single-image vignetting correction. *PAMI*, 31(12):2243–2256, December 2009.

[120] Z.H. Zhou and X. Geng. Projection functions for eye detection. *Pattern Recognition*, 37(5):1049–1056, 2004.

[121] Maria Zontak and Israel Cohen. Defect detection in patterned wafers using anisotropic kernels. *Machine Vision and Applications*, 21(2):129–141, 2010.