Impact of Object Extraction Methods on Classification Performance in Surface Inspection Systems

Stefan Raiser, Johannes Kepler University of Linz

Edwin Lughofer (Corresponding Author: edwin.lughofer@jku.at)

Johannes Kepler University Linz,

Christian Eitzinger, PROFACTOR GmbH

Comment: Paper for Special Issue on 'Integrated Imaging and Vision Techniques for Industrial Inspection'

August 26, 2008

Abstract

In surface inspection applications, the main goal is to detect all areas, which might contain defects or unacceptable imperfections and to classify either every single 'suspicious' region or the investigated part as a whole. After an image is acquired by the machine vision hardware, all pixels, which deviate from a pre-defined 'ideal' master image are set to a non-zero value, depending on the magnitude of deviation. This procedure leads to so-called contrast images, in which accumulations of bright pixels may appear, representing potentially defective areas. In this paper, we present various methods, ranging from classical image processing techniques to machine-learning based clustering approaches, for grouping these bright pixels together into meaningful objects. One important issue here, is to find reasonable groupings even for non-connected and widespread objects. Generally, these objects correspond either to real faults or to pseudo-errors, which do not affect the surface quality at all. Assuming that objects have to be extracted correctly for a proper classification, we also study the impact of the different extraction methods onto the classification accuracies of classifiers which are trained from feature vectors calculated for the objects found, based on labelled data from the user. In our investigation we will consider artificially-created contrast images as well as real ones, recorded on-line at a CD imprint production and at an egg

inspection system.

Keywords surface inspection, contrast images, object extraction, clustering, image classifiers

1 Motivation

Machine vision systems currently form an integral part of many machines and production lines. They perform different kinds of monitoring or inspection tasks, without the need of actually touching the parts under investigation. The availability of powerful, yet cheap, cameras and computer systems has made a 100% inline quality inspection instead of random examination economically feasible.
A typical application of machine vision in industry is surface inspection, where the surface of a discrete part or endless material is checked for imperfections such as scratches or ridges. In this connection, any deviations from a desired characteristics have to be identified. Surface inspection is of great importance in automobile, electronics, semiconductor, metal-working and printing industry in order to assure a high product quality and hence to decrease customer complaints and additional costs for re-production.

In this sense, methodologies and methods for surface inspection problems were developed during the past years, for an overview see for instance [26]. In [21] [16] [12] particular solutions of inspection in textile products are demonstrated, which are based on an analysis of textures by classical image processing methods. In [13] an approach for surface inspection for wooden material (classification into red oak boards, classes of surface defects, and one class of clear wood) is demonstrated, where image pre-processing techniques are applied for separating dark and bright regions and where the surface defects are characterized by the mean of the gray levels and CAR model parameters. In [14] detection and classification of structural defects on silicon carbide (SiC) wafers is presented, where a specific fault class (so-called micro-pipes) is under the major investigation. Most of these methods have one in common: they were developed for very specific application scenarios resp. mostly for one specific industrial inspection system (or even specific fault classes) and are not directly applicable for a wider range of problems. This also means that the whole development effort for one inspection system has to be taken again and again and can be quite huge. Some other approaches such as [10] [15] exploit machine learning methods (such as self-organizing maps or coupled hidden Markov models) for surface

inspection and are shown to be applicable for concrete application scenarios (e.g. inspection of wooden surfaces or steel surfaces containing three-dimensional flaws [18]).

We propose a surface inspection framework (shown in Figure 1), which is applicable to a wider range of surface inspection problems as it includes a combination of multiple generic components such as machine learning methods. The only assumption therein is that a so-called contrast image is available.

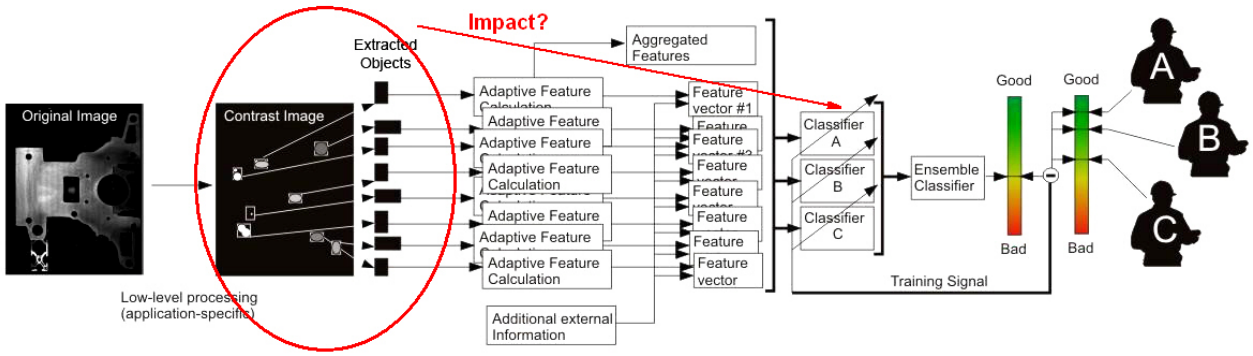


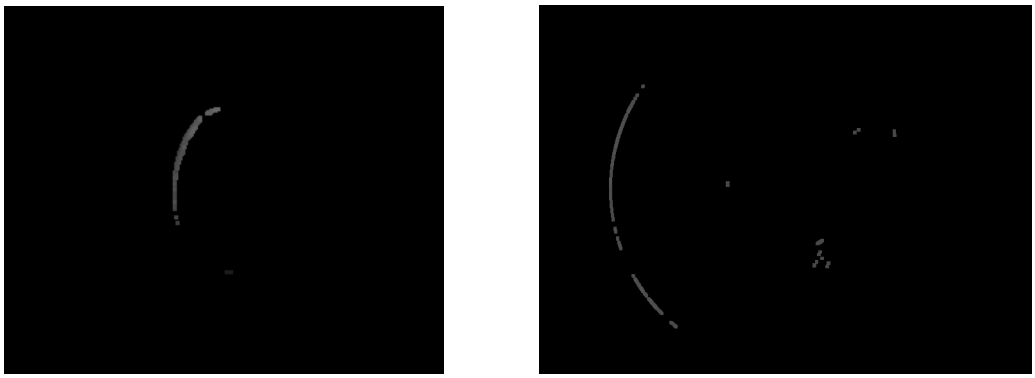
Figure 1: Surface inspection framework; the object extraction component in contrast images surrounded with an ellipsoid

Generally, this image is calculated from the original image of the workpiece to be inspected by applying specific, low-level image processing routines. All regions, which differ from a perfect master part, are set to a non-zero value, depending on the magnitude of deviation. Someone may now assume that, when a certain contrast image contains any (significant) deviation pixel(s) at all, it always shows a defective production item, which has to be classified as 'bad'. However, in most applications the correct classification depends on the distribution of the deviation pixels and the characteristics of their local accumulation(s), as so-called 'pseudo-errors', which do not correspond to real faults, may occur (see also Section 2). As a first step all 'suspicious' objects, which represent potential defects, are extracted from the contrast image. Then, for each of these extracted objects, features, i.e. area, homogenity and roundness, are calculated (object features). These single features are combined in order to characterize the whole bright pixel area in the image (aggregated features), i.e. their number, average brightness and density. In a training (off-line) phase, these features are calculated for a set of stored images, which are already classified by a human expert, and, based on this information, a classifier is trained using machine learning methods. In the extraction (on-line) phase, the trained classifier automatically assigns each new image to the best-matching class based on the calculated features and finally generates a 'good'/'bad' decision for the currently inspected part.

The main contribution of this paper is now to describe various methods for extracting 'suspicious' objects in the contrast images (as done in Sections 3 and 4) and to analyze the impact of these methods on the predictive performance of the image classifiers (Section 5). The latter gives us a kind of measure, how well the object extraction methods perform in identifying the fault-candidate objects. We will analyze this impact based on three data sets, one artificial data set including specific rules mimicking typical errors at surface inspection systems, and two data sets from real-world inspection problems, eggs and CD imprint inspection. We will apply three different types of classification methods for image classifier training in order to eliminate an eventual bias of a classifier when coupled with a certain object extraction method and so to see whether one approach can be favored over others.

2 Problem Statement

All non-zero pixels in the contrast image correspond to regions, which differ from the ideal appearance, and the amount of deviation is reflected by the pixel's values. In most cases, the contrast image can be visualized as an 8-bit or 16-bit gray scale image. Due to the fact, that all deviating areas are detected, the bright pixels can represent either real defects or pseudo-errors, i.e. artifacts, which emerge during the computation of the contrast image caused by varying illumination conditions or improper image alignment.



(a) (b)

Figure 2: Contrast images containing real defects and pseudo-errors from a) an egg inspection, b) a CD imprint inspection system.

Figures 2a and 2b show contrast images from two different surface inspection applications: an egg inspection and a CD imprint inspection system. In these examples, most pixels are black, whereas the

deviating regions consist of rather bright gray values. In both cases arc-type structures occur, which represent pseudo-errors, while the other non-zero pixels result from real faults.

In order to further process the information contained in the contrast image, the bright pixels have to be grouped together into meaningful objects, ideally separating pseudo-errors from real ones. These distinct regions are then returned individually to the next processing steps, i.e. the feature extraction. Typically, the objects are characterized by forming a connected set of pixels. Two pixels are considered to be connected, if they are next to each other on the rectangular pixel grid a digital image consists of. If the diagonal adjacent pixels are excluded it is called 4-connectivity, if all 8 neighbors are considered, it is called 8-connectivity. Every set of pixels that are connected according to either of these two definitions is called a connected component. But in a variety of surface inspection environments, non-connected objects can occur.

(a) (b)

Figure 3: Contrast images containing disconnected objects from a) an egg inspection, b) a CD imprint inspection system.

Figure 3a shows the contrast image of an egg covered with disconnected spots of yolk (a fault case). An example of highly-scattered objects occurring during CD imprint inspection is shown in Figure 3b. Generally speaking, object extraction methods have to be able to find correctly all objects, which consist of bright pixels that visually belong together, but need not necessarily be connected. For a human it is quite easy to find the natural groupings in the contrast image, because he or she is able to interpret the image in a 'bigger context' and can perceive patterns, even if they are very fragmented. Automatic procedures are challenged by the fact, that they have to deal with connected and non-connected objects of arbitrary size, shape and number without any further information on the image content. In this sense, in order to match the human interpretation of an image (reflected in his/her labels), the object

extraction method should approximate the human grouping as close as possible. Otherwise, the features, calculated from the objects automatically found, will be misleading for a classifier, which is trained to make human-like decisions. Feature values may get disturbed and probably reflect other classes than they really belong to.

In the next sections, first we will outline a very common object extraction approach, namely the connected component algorithm, together with some enhancements and then give an overview on machine learning approaches based on clustering techniques.

3 Common Techniques for Object Extraction

In our framework from Section 1, the contrast image, which i.e. is calculated by taking the absolute difference between the current image and a fault-free master image, already contains all interesting areas as bright pixels. So, the following object extraction step now has to find meaningful groupings for the bright pixels, which reflect the underlying object structure as good as possible.

In this paper, we want to investigate the issue of grouping. Thus we do not consider any textural structure of the defects and just use a binary image. [1], generally only the binarized version of the image is used for object extraction. In the simplest case, a global thresholding is applied on the contrast image, which sets all pixels with a gray value above a certain level to one and the remaining pixels to zero. So, the thresholding produces a binary image with a black (0) background and a white (1) foreground, which contains the objects of interest. A detailed description of various thresholding methods can also be found in [6].

The most common approach to obtain the individual objects from a binary image is the connected component algorithm (CC) ([23]). It assumes, that the objects to be found consist of a connected set of pixels (as defined in the previous section). So, all connected components detected by the algorithm correspond to the distinct objects. In literature, various implementations of the connected component algorithm are discussed. The outcome can either be a list of sets of connected pixels or a colored image (aka labeled image) with different colors assigned to the distinct objects.

In Figure 4 the results of the connected component algorithm, applied on the binarized contrast images from the previous section, are shown. In all cases, the number of objects found exceeds the

[1] Since a single object can consist of pixels with different gray values, this information can be even misleading.

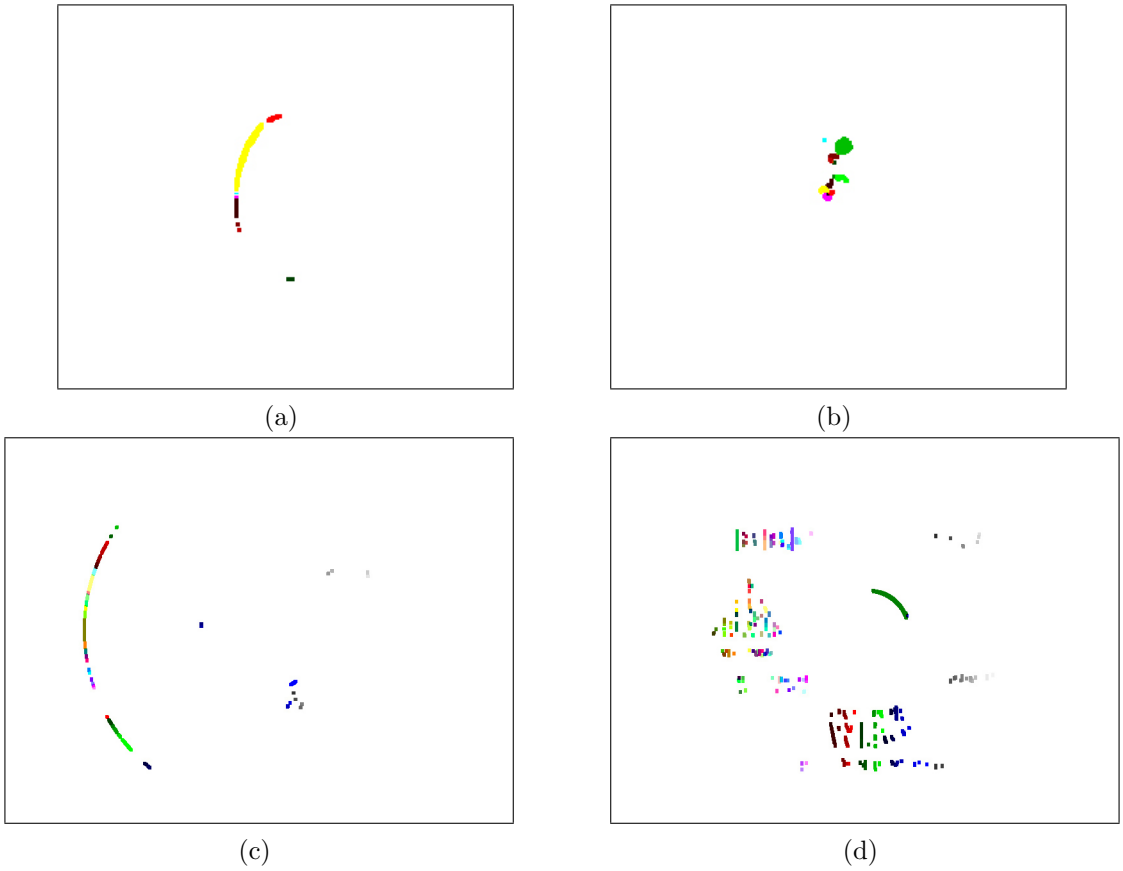


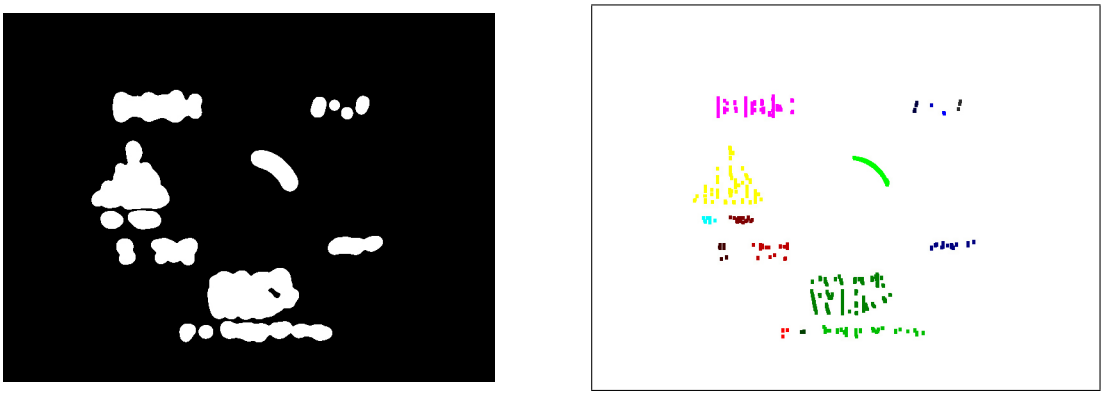
Figure 4: Results of connected component labeling with a) 8, b) 11, c) 51, d) 203 objects found.

number of objects that a human would perceive/identify, because the assumption of connectedness is not fulfilled. Especially, for images with the highly-scattered 'pixel-clouds' (see i.e. Figure 4d), the algorithm returns far too many objects, which leads to erroneous feature values and consequently may deteriorate the performance of the classifiers.

So, in the case of non-connected objects, an algorithm is needed, which somehow groups pixels together on a scale, bigger than the pixel neighborhood, in order to reduce the number of objects found. One traditional way is to improve the connected component algorithm by performing a pre-processing step before applying it.

A simple way to enhance the binarized contrast image with respect to the outcome of the connected component algorithm, is to blur the image with i.e. a gaussian lowpass filter in order to close gaps between bright pixels. Then the blurred result is binarized and the connected component algorithm is applied on it. As the connectedness between the pixels has increased due to the blurring, the number of objects found is generally much smaller than without filtering. Finally, the pixels generated while blurring the contrast image, are removed again from the labelled image, while keeping the labelling

constant. Although in many cases the result is quite good, it strongly depends on the choice of the blurring filter and its parameter settings.



(a) (b)

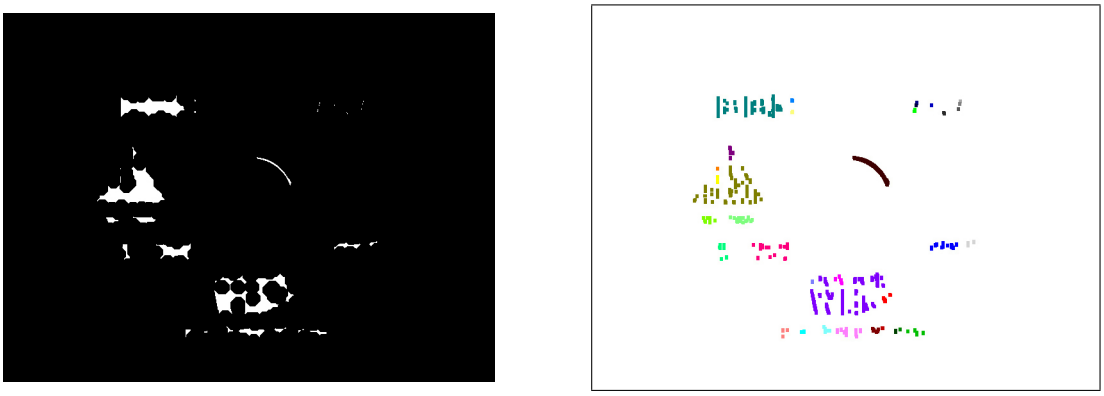
Figure 5: a) Blurring with gaussian filter, b) Result of connected component labelling (16 objects found).

Figure 5a shows a CD imprint contrast image, which was binarized, then blurred by a gaussian lowpass filter (with σ=4.4 and a kernel size of 30x30 pixels) and finally binarized again by global thresholding. After applying the processing steps outlined above, the final result (Figure 5b) exhibits a much smaller number of objects.

Another way to increase the connectedness between bright pixels, which are close together, yet not connected, is by applying morphological operations as a pre-processing step on the binarized contrast image. In order to remove holes in the white pixel areas and small gaps between these areas, a morphological closing can be performed, consisting of a dilation followed by an erosion with a structuring element, i.e. a disk with a certain size (see [6]). After that, the connected component algorithm is applied, hence extracting a smaller number of objects. Finally, the additional pixels generated by the closing are removed again, keeping the labeling constant. The result strongly depends on the choice of the structuring element, its shape and size respectively. Even small changes can produce a different labelling and a varying number of objects found.

Figure 6a illustrates the effect of closing the CD imprint contrast already shown before with a disk-shaped structuring element. Again, the final result (Figure 6b) is improved by the preprocessing.

As a conclusion it can be said, that the connected component algorithm alone is (obviously) not a good choice for images containing non-connected objects. Performing various kinds of preprocessing, leads to more reasonable results, but depends strongly on the parameter setting of the preprocessing algorithm(s) and requires additional computational effort. In this sense, we exploit specific clustering-



(a) (b)

Figure 6: a) Morphological closing, b) Result of connected component labelling (35 objects found).

based approaches which circumvent these shortcoming under specific conditions as demonstrated in the subsequent section.

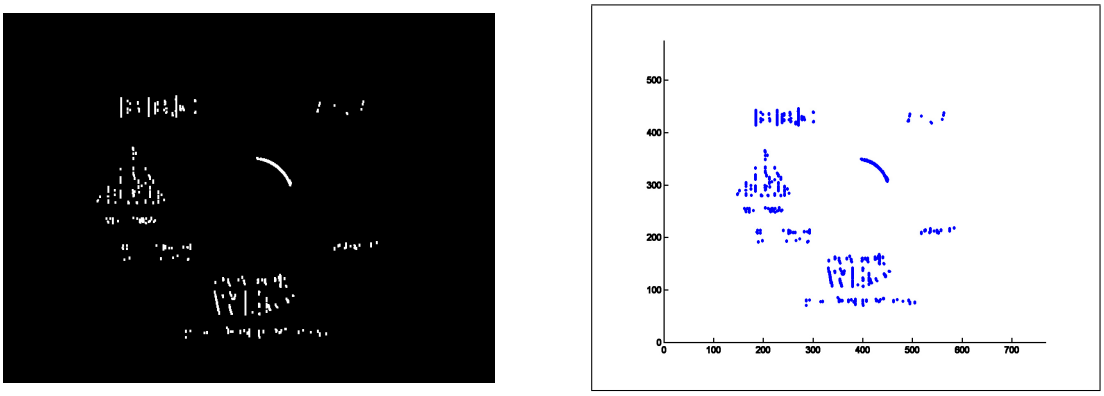
4 Clustering-Based Approaches for Object Extraction

Clustering methods have been used in a variety of disciplines leading from statistics and numerical analysis to data mining and machine learning. Generally speaking, clustering can find 'natural' groupings (clusters) in data. Each of these clusters consist of data points that are similar between themselves and dissimilar to those of other groups with respect to a previously chosen similarity/dissimilarity measure. A comprehensive description of clustering can be found i.e. in [11] or [5].

This characteristic makes clustering a candidate for solving our object extraction task, as usually different types of fault candidates appear in different groups of non-zero pixels in the contrast image. The basic idea is to consider the white pixels in the binarized contrast image as data points, which serve as input for the clustering algorithm. Figure 7a shows the binary version of a CD imprint contrast image and Figure 7b is a visualization of the corresponding data matrix, illustrating the conversion of pixels into two-dimensional data points.

In this case, the original task, namely the extraction of objects, becomes equivalent to the problem of finding clusters in the set of data points. The question, which pixels belong together or to the same object respectively, is now addressed by the clustering algorithm and on how it partitions the data points.

How clustering methods can be integrated in the object extraction process is shown in Figure 8.



(a) (b)

Figure 7: a) Binarized CD imprint contrast image, b) White pixels as data points for clustering.

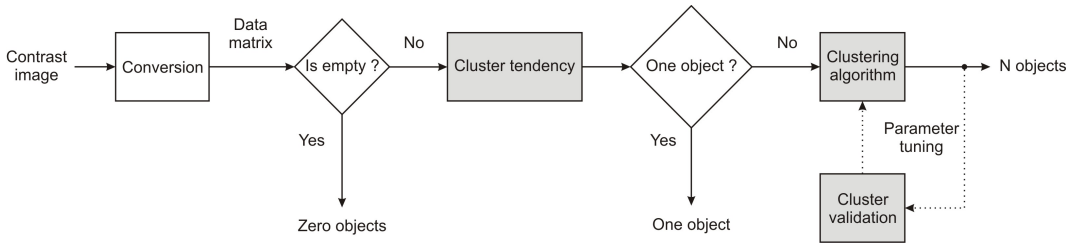


Figure 8: Integration of clustering methods in the object extraction process.

After the binary contrast image has been 'converted' into a data matrix, a cluster tendency analysis checks for the presence of a clustering structure. If the image contains more than one object, a clustering algorithm is applied on the data points in order to extract the individual objects. Finally, (optional) cluster validation techniques can be used to tune the parameters of the clustering algorithm.

4.1 Cluster tendency

Before applying a clustering algorithm, various tests, that indicate whether the available data possess a clustering structure at all, can be performed. These methods examine the data to see if there is any merit to a cluster analysis or not. In the image processing context, cluster tendency addresses the question, whether there is more than one object in the image. In the case of a single object, clustering is not really useful. All foreground pixels can be grouped together and the object extraction is finished. A very intuitive cluster tendency test based on nearest neighbor distances is the calculation of the Hopkins index h (aka Hopkins test) as described by [9] and [19]. After choosing m random points s_m from the data matrix and also m random points r_m from its convex hull, the distances d_{s_i} (d_{r_i}) from

each s_i (r_i) to its nearest neighbor is calculated and inserted in the following formula:

$$h = \frac{\sum_{i=1}^{m} d_{r_i}^2}{\sum_{i=1}^{m} d_{r_i}^2 + \sum_{i=1}^{m} d_{s_i}^2} \in [0, 1]$$

The value of h lies between zero and one, whereas the three typical cases shown in Figure 9 can be distinguished:

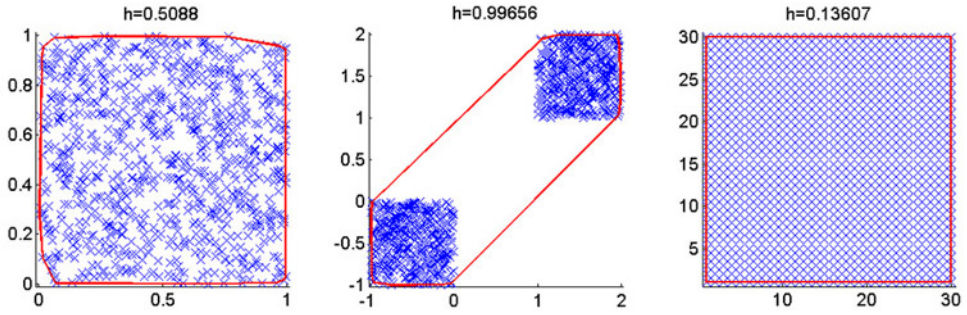


Figure 9: Three data sets and their Hopkins indices.

1. If $h \approx 0.5$ (as seen on the left), then the data points are likely to have a random pattern (only a single cluster).

2. If $h \approx 0$ (as seen on the right), then the data points are located on a regular grid (with as many clusters as data points).

3. If $h \approx 1$ (as seen in the middle), then a clustering structure exists (with a number of clusters somewhere between 2 and the number of data points).

In the context of object extraction, the Hopkins index h can be used as a stand-alone test for cluster tendency. If h is smaller than a certain threshold[2], all white pixels belong to the same (single) object and no clustering has to be done.

The performance of the Hopkins index in detecting images with single clusters was tested based on artificial data sets where the real number of objects in the images were known (as generated artificially but simulating a real-world inspection process). For each image, containing more than 20 white pixels,

[2]Threshold values between 0.5 and 0.8 showed good experimental results here.

the index was calculated[3]. If its value exceeded a threshold of 0.6, the image was considered to contain more than one object. Otherwise, all white pixels in the image were grouped together as a single object. The results were remarkable good as 1.) a high percentage (above 85%) of single clusters could be identified, whereas the other 15% were recognized as single clusters by the clustering algorithm, afterwards (so the 15% miss-detection were compensated by clustering) and 2.) a very low percentage (smaller than 1%) of images were miss-detected as single clusters (as they contained more than one cluster) and hence not sent into the clustering algorithm. This causes a disturbance value of 1% in the features, which is fortunately a quite low value.

4.2 Clustering algorithms

After checking the existence of clustering structures, or in other words, after having verified that there exists more than one object in the binary image, a clustering algorithm can be applied in order to find the natural grouping of the foreground pixels.

For the object extraction task, a clustering algorithm is needed, which can detect clusters of arbitrary shape, because the foreground pixel areas in the binary image can exhibit any form. After the grouping, each foreground pixel should be assigned to a distinct cluster (object), which means (in terms of clustering), that a crisp partition is needed as output. Also, if the algorithm returns a hierarchy of partitions, a cutoff level has to be determined in order to get a single partition.

The following clustering algorithms are all capable of finding arbitrarily shaped objects in a binary image and their performance has been investigated for the example applications described in this paper:

- **Single linkage hierarchical clustering (HC-SL)** [25] is a representative of the agglomerative hierarchical algorithms. It starts with every data point being a cluster and then iteratively merges the closest pair of clusters according to some distance measure, until all data points are in one cluster. In the single linkage version of the algorithm, the distance between two clusters is measured by the distance of the closest pair of data points from each cluster. This kind of distance measurement enables the algorithm to identify clusters of arbitrary shapes and different sizes. Also elongated objects can be detected due to the so-called 'chaining-effect', which groups lines of distinct, yet closely connected data points together. For these reasons the HC-SL algorithm

[3]If too less data points are used for the calculation of the Hopkins Index, its value has no significance.

is quite feasible for object extraction. But as this task demands a single partition instead of a hierarchy, the hierarchical algorithm has to be stopped at a specific level of the hierarchy, when the desired partition is obtained and further grouping only would combine clusters, which do not belong together. The stopping criterion is implemented as a threshold , called the $cutoff$ value, which determines the maximum distance allowed for merging two clusters.

- **DBSCAN** [4] is a density-based algorithm, which regards clusters as dense regions in the data space, separated by regions of lower density. At the beginning, the algorithm initializes all data points as unclassified. Then it starts with an arbitrary data point and finds all points that can be reached from it, while staying inside a region of high data density. If the point turns out to be somewhere inside a dense area, then a cluster is formed. If the point is at the border of a dense area, the algorithm proceeds with the next unclassified point. If the point is neither inside nor at the border of a dense area, it is classified as noise. This continues until all of the points have been processed. The DBSCAN algorithm is optimized to work efficiently on large spatial databases, which makes it especially useful for object extraction, as the number of bright pixels in the contrast image can be quite large.

- **Reduced Delaunay graph (RDG)** [17] represents a graph-based approach and is based on partitioning the Delaunay graph of a given data set. All edges of the graph are weighted according to a normalized distance measure and clustering is performed by removing all edges for which the weight is larger than a fixed threshold. Finally, each connected component of the remaining graph corresponds to a single cluster. According to the authors, it is able to find clusters of complex shape and groups data points in a similar way as human observers.

- **Normalized cut (NCUT)** [24] is a spectral clustering method, which represent a very powerful class of clustering algorithms arised at the early 2000's. It starts with a fully connected graph, for which each node represents a data point and the edge between two points is weighted by their similarity, i.e. their distance from each other. The algorithm recursively finds the bi-partitioning of the graph, which minimize a certain criterion, namely the normalized cut value. The recursion stops, if the quality criterion for the cut becomes higher than a specified threshold or there are only single points left.

4.3 Parameter Setting and Cluster Validation

Clustering, as well as other object extraction approaches, has at least one input parameter, which significantly influences the number of objects found by the algorithm. As in general, every image contains a different number of objects, it is not sufficient to determine values for these critical parameters by a trial-and-error method applied on a small set of test images. Fortunately, in many cases heuristic methods or estimation formulas are available to set the parameters to reasonable values for each new image to be clustered.

In the following, the essential parameters of the clustering algorithms from the previous section are listed and some 'practical' methods for the determination of these parameters are proposed:

- **Single Linkage Hierarchical Clustering (HC-SL).** The $cutoff$ value, used to select a single partition from the hierarchy, determines the number of objects found. In the context of image processing, it is feasible to make the $cutoff$ value depending on the size of the image, because the size gives a strong hint on the scale of the objects, which the image contains[4]. Here, a typical rule of thumb is to set the $cutoff$ to

$$cutoff = \frac{\#imageRows + \#imageColumns}{30}.$$

  As its value can be determined a-priori according to the formula, the algorithm can be stopped, when the distance between the closest clusters exceeds the $cutoff$. By contrast, other methods for finding an appropriate $cutoff$ value (i.e. as described in [20]) need the whole hierarchy to be computed, which leads to a longer computation time.

- **DBSCAN.** In this case, the setting of two parameters, namely $MinPts$ and ϵ, which relate to the expected density of the 'thinnest' cluster, is required. $MinPts$ can be fixed at 4 as proposed by [4] for two-dimensional data. The value of ϵ can either be determined by using the *sorted k-distance graph* (see also [4]) or the following estimation formula (taken from [3]):

$$\varepsilon = \sqrt{\frac{(\max\limits_{1\le i\le m}(x_m) - \min\limits_{1\le i\le m}(x_m)) \cdot (\max\limits_{1\le i\le m}(y_m) - \min\limits_{1\le i\le m}(y_m)) \cdot MinPts \cdot \Gamma(2)}{m \cdot \pi}}$$

[4] In surface inspection applications usually the object to be investigated is almost completely filling the image.

with the bright pixel coordinates stored in a $m \times 2$ matrix

$$D = \begin{bmatrix} x_1 & y_1 \\ \vdots & \vdots \\ x_m & y_m \end{bmatrix}.$$

- **Reduced Delaunay Graph (RDG).** The threshold for the edge weights, applied when removing edges from the Reduced Delaunay Graph, influences the amount of connected components and hence the number of objects. When sorting all edge weights of the graph in a descending order and making a plot, with the x-axis leading from one to the number of edges and the y-axis showing the sorted weight values, an L-shaped curve evolves. A good threshold value can be obtained by taking the edge weight at the 'knee' of this curve. It can be determined automatically by normalizing the two axis and finding the point with the closest (euclidian) distance to the origin. Figure 10 illustrates the procedure on an artificial sample image.

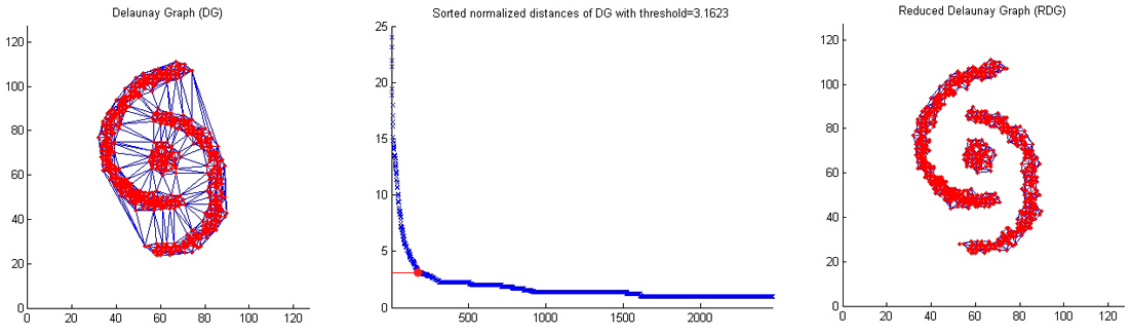


Figure 10: left: Delaunay Graph, middle: Threshold Determination, right: Reduced Delaunay Graph.

- **Normalized Cut (NCUT).** As the algorithm starts by constructing a weighted graph with the weights generally calculated using a gaussian similarity measure, the most important parameter is the value of σ in the gaussian function, since it influences strongly the neighborhood relationships and hence the number of clusters found. In case of object extraction, meaningful values for σ are depending on the image/object size. So it is feasible to fix its value at some percentage (typically between 5 and 10 percent) of the image dimensions. The second parameter, which is the threshold for $Ncut$, used as stopping criterion for the recursion, also has to be set by the user. In practice, the exact value is not very critical and typically somewhat above zero.

All clustering results presented in this paper have been generated by using these automatic parameter determination methods.

Another approach is to use so-called cluster validation techniques (see [7]) to determine an appropriate parameter setting or to perform an automatic fine-tuning of estimated parameter values during on-line operation. Here, the most feasible methods for automatic parameter tuning are the ones based on relative criteria, as they offer the possibility to choose the best clustering scheme out of a set of schemes according to some criterion, reflected by a so-called cluster validation (CV) index. After clustering with different parameter settings, the resulting partitions are rated on-line on the basis of the CV index and the 'best' partition is selected and returned as final result. As this iterative approach demands high computational effort, it is not further discussed here.

4.4 Visualization Results on Real-World Image Sets

For the real image dataset, no expert knowledge has been available about the objects actually contained in the images, so the assessment of the object extraction quality had to be carried out simply by visual judgement. Figures 11 and 12 illustrate the actual grouping of the white pixels according to the different approaches for images from a CD imprint inspection [5]. As the objects in this application are highly discontinuous and scattered, all clustering methods clearly outperform the connected component algorithm and return quite meaningful groupings.

5 Impact of Object Extraction on Classification Accuracy

In this section, we study the impact of the various object extraction techniques as outlined in the previous sections onto the accuracy of image classifiers trained in surface inspection systems for distinguishing between images showing bad and good production items. As already shown in Section 1, the image classifiers are built into a whole image classification framework (Figure 1), consisting of several components, including an object extraction and feature calculation part. Hereby, features are calculated

[5] The colors, used to mark the objects found, have no meaning and can differ between the clustering algorithms even when the objects have been extracted identically.

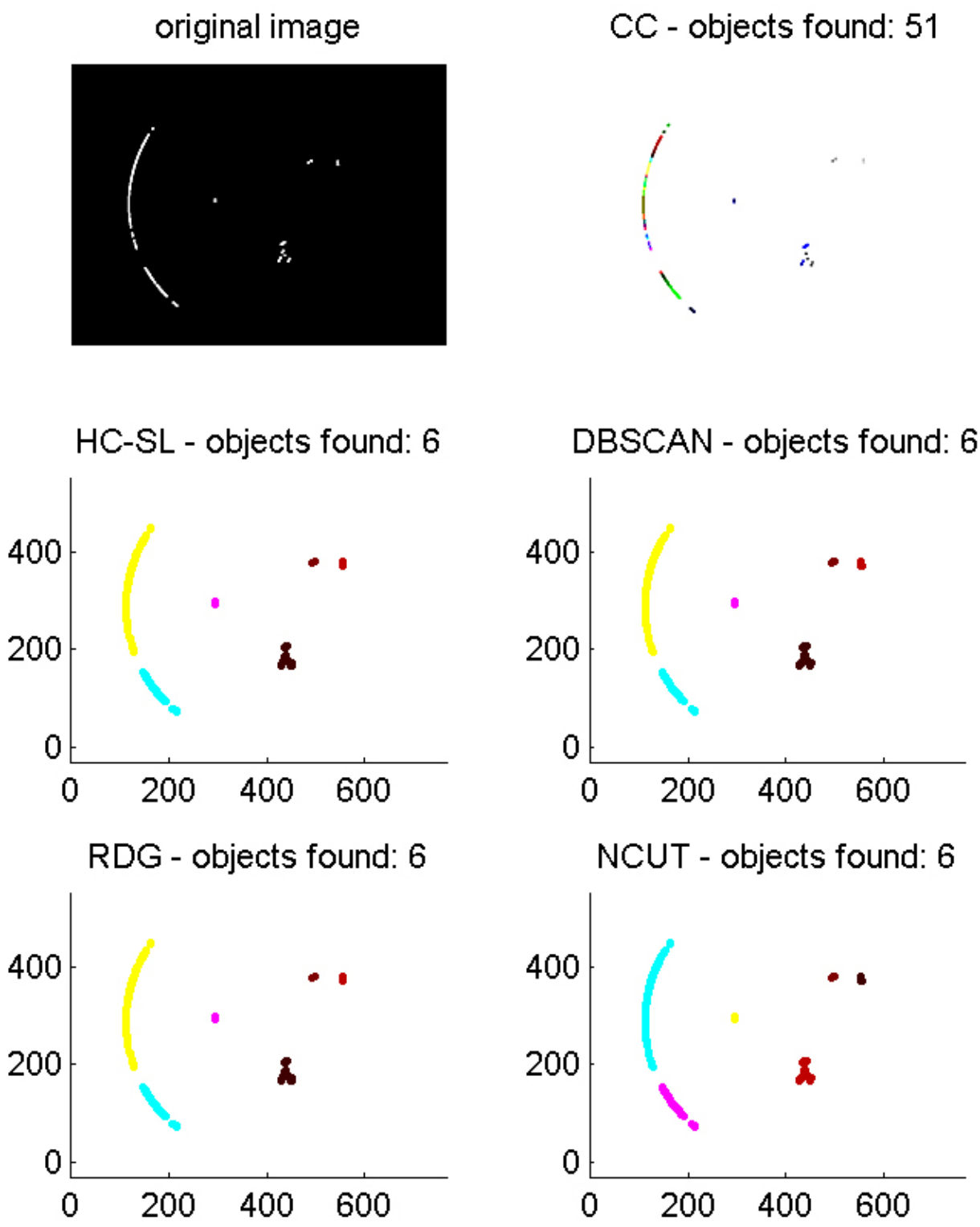


Figure 11: First CD imprint inspection example: Here, an image with an arc-type structure and four small clusters is shown. All clustering algorithms agree in finding six distinct objects. Although the arc is split in two pieces, the results are very good compared to the outcome of the connected component algorithm, which finds too many objects.

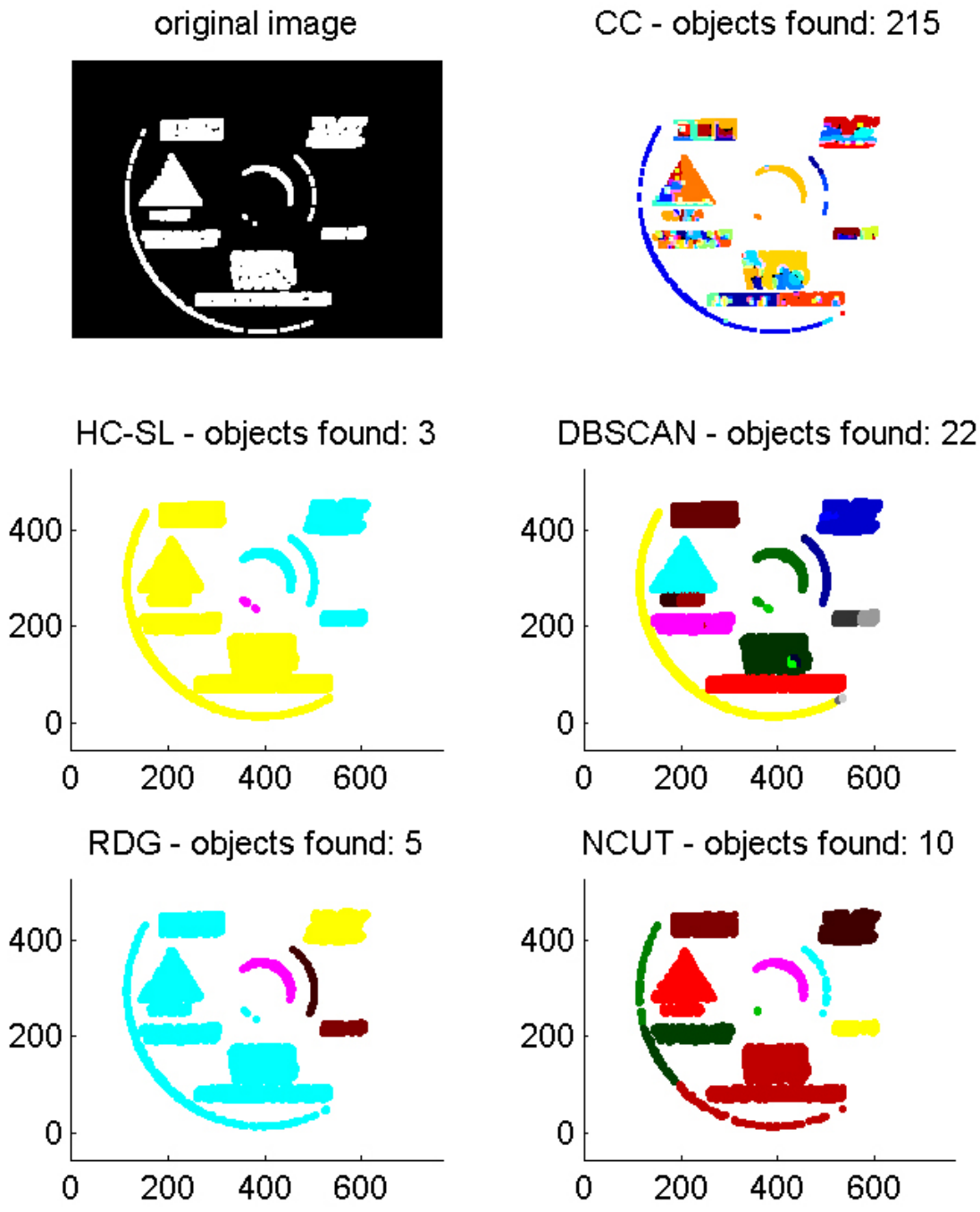


Figure 12: Second CD imprint inspection example: In this case, clustering algorithms perform dissimilarly: again, the connected component algorithm returns worse results, but also HC-SL is quite weak as connecting too much objects together and delivering to big objects. DBSCAN clearly over-estimates the number of objects (some compact objects are divided up into more), whereas RDG also under-estimates it (as HC-SL). Intuitively normalized cut performs best, except for the longer arc-type object at the left hand side which is divided up into three objects

No.	Description	No.	Description
1	Number of ROIs	10	Max. grey value in the image
2	Average minimal distance between two ROIs	11	Average grey value of the image
3	Minimal distance between any two ROIs	12	Total area of all ROIs
4	Size of largest ROI	13	Sum of grey values of all faults
5	Center Position of largest ROI, x-coord	14	Maximal local density of the ROIs
6	Center Position of largest ROI, y-coord	15	Average local density of the ROIs
7	Max. intensity of all ROIs	16	Average area of the ROIs
8	Center Position of ROI with max. intensity, x-coord	17	Variance of the area of ROIs
9	Center Position of ROI with max. intensity, y-coord		

Table 1: List of aggregated features used. These features have been chosen because of their relevance for a very wide range of surface inspection applications.

from extracted objects, whereas we label the pixels according to the objects they belong. We distinguish between two types of features: object features and aggregated features. The former are characterizing single objects by their size, shape, density, statistical values from gray level histograms etc. Here, it is quite intuitive that an object extraction approach which does not correctly group the objects of interest (for instance two or more objects are connected to one which should not be connected resp. too much objects are extracted – as often the case when using connected components, see figures in previous section), affects somehow the value of the features extracted from these objects. We call this affect a kind of disturbance of feature vectors in the feature space. This disturbance is assumed to guide the classifier to a wrong solution in the sense, that it does not represent the correct implicit induction of classes, previously assigned by the experts to the real objects (experts usually know how the objects look like). The disturbance can reach a level, such that the object feature vector belonging to an object which is labelled as class X has similar values than another object feature vector (perfectly extracted) belonging to an object which is labelled as class Y. The aggregated features are describing the characteristics of images as a whole and hence are used for training an image classifier for classifying images into 'good' and 'bad' ones. They implicitly contain information from the single objects and hence may also be 'disturbed' by bad object extraction approaches. The complete list of aggregated features we used is listed in Table 1.

Furthermore, if single objects are not labelled, an aggregation or a clustered information of object features is carried out [2] in order to include the nature of the objects in the whole image classifier.

We applied an empirical evaluation of image classifier accuracies on three different data sets for showing the impact of the various object extraction approaches onto the classification performance: an artificial data set where good/bad decision rules at a real industrial surface inspection are imitated, an on-line recorded data set at a CD imprint production process and a data set from an egg inspection system.

5.1 Characteristics of the Data Sets

5.1.1 Artificial Surface Inspection Data Set

We used five sets of artificial test data, each with 20.000 images, which were labelled automatically either as good (accept) or as bad (reject) with about 10.000 images in each class. No labels on the single objects were given. In order to generate the labels, a set of rules was used for each set of test images. The rules were based on descriptions that are regularly found in quality control instructions, such as "part is bad, if there is a fault with size > 1.5 mm". The rules also included more complicated combinations, such as "part is bad, if there is a cluster of 4 faults, each with a size > 0.75mm". Three to five such rules were logically combined for each set of images. The images and the rules were chosen to have some resemblance to inspection of machined parts. The first two data sets (ArtifData01 and ArtifData02) contained quite simple rules on the whole image, where the other three data sets (ArtifData03-05) contained more complex rules which also depend on the distribution of the objects in an image. Two examples of artificial images (size 128 time 128 pixels) are shown in Figure 13, the left image shows a bad one, the right image a good one, even including some pixels marking significant deviations.

5.1.2 Egg Inspection

In this real-world application, hen's eggs are inspected in order to identify dirt and yolk, which might cover parts of the eggshell. The images have a size of 313x262 pixels and are gray scale. There are three sets with 4342 images available: one with dirt only, one with yolk only and a third one with both kinds of defilement. In this sense, it was possible for us to label the object feature vectors according to dirt and yolk. The images were pre-labelled by the experts with 'good' and 'bad', whereas most of the images were labelled as good, so for classifier training the input was a quite imbalanced data set. Due to the nature of dirt and yolk, their shape can be arbitrary. The correct number of objects in the images is

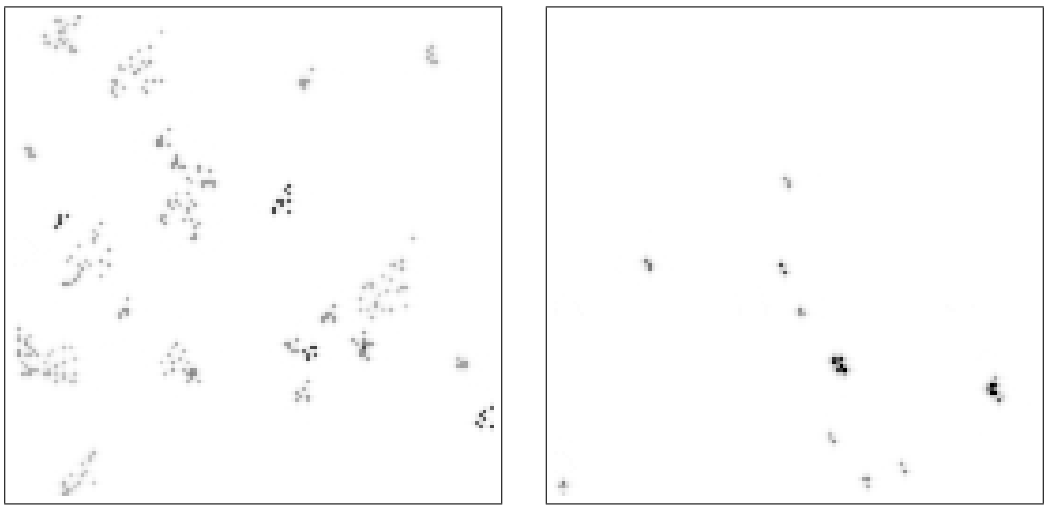


Figure 13: Left: artificial images labelled as bad, right: artificial image labelled as good

unknown, such that an object extraction approach supporting the extraction of objects with arbitrary shape is required. Example deviation images from the egg inspection system are shown in Figures 2a 3a, 4a and 4b.

5.1.3 CD Imprint Inspection

The images in this application example come from a CD imprint inspection process. In order to guarantee the quality of the imprints, a color matrix camera is installed in the production line, which takes an image of each CD. These images are compared with the image from the corresponding fault-free master CD ($\pm$ a threshold) and each deviation is marked as a probable defect. The outcome is a gray scale image of size 768x576 pixels, where the gray levels correspond to the amount of deviation from the master. Obviously, the defective areas have arbitrary size and shape and often exhibit some kind of scattered structures, i.e. like they occur in the case of ink splashes. The major effort was to distinguish between real defects such as a color drift during offset print, a pinhole caused by a dirty sieve ($\rightarrow$ color cannot go through), occurrence of colors on dirt, palettes running out of ink so-called pseudo-errors, like for instance shifts of CDs in the tray (disc not centered correctly), causing longer arc-type objects (e.g. see left side of Figure 11) or masking problems at the edges of the image or illumination problems ($\rightarrow$ causing reflections). The CDs showing the latter artefacts should be recognized as 'good'. After object extraction, features are calculated for the objects found. They serve as input for the classifier, which decides whether the imprint is okay or has to be rejected.

5.2 Experimental Setup

While for the artificial data sets the good/bad labels of the images were automatically given by the predefined rules, the egg data sets containing 4342 images were labelled by one expert at the system and the CD imprint data set containing 1534 non-black images (purely black images can be trivially identified as fault-free) were labelled by four different operators with a slightly different view on the objects and whole images. As done by real experts, the labelling is representing the characteristics of good and bad images quite well. In this sense, we assume that wrongly identified objects will have an influence on the classification accuracy of the classifiers built up based on the expert labels, as feature vectors extracted from the objects are disturbed. The number of object features in sum are 57, the number of aggregated features (characterizing images as a whole) are 17 (as shown in Table 1). For the artificial and CD imprint data sets we used an unsupervised information from the objects (by aggregating their feature values among each image) and appended them to the aggregated features. For the egg data set we used directly the object labels for training an object classifier distinguishing between dirt and yolk. According to an interpretation of the no free lunch theorem by Wolpert [33], which says that for different data sets different types of machine learning methods (and also classifiers) perform best among others, we do not stick to one fixed classification method, but we exploit three different well-known classification methods, namely decision trees generated by the CART approach [1] (implementation in MATLAB's statistics toolbox), support vector machines [29] [22] (lib-SVM implementation) and possibilistic neural networks [32] (implementation in the MATLAB's neural network toolbox). This has also the effect that we see whether and how the different object extraction approaches affect a variety of different classifiers in terms of predictive accuracy. When we assume that connected components generates a quite high disturbance of the feature vectors (as usually producing by far too much objects as these are not necessarily connected, see Figures in previous sections), this also gives us a kind of 'robustness' analysis with respect to noise for the various classification methods. For all classifiers we apply a 10-fold cross-validation step [27] coupled with a best parameter grid search scenario and elicit their errors, as these can be seen as a good estimation of the expected prediction error on new unseen samples, see [8]. The search over a parameter grid is for eliciting the optimal parameter setting of each single classifier (optimal in terms of minimizing the CV error), which is the pruning level for CART, C and γ for SVMs and the spread of the neurons for possibilistic neural networks.

CART	CC	HC	DBSCAN	RDG	NCUT
Art. #1	87.86	87.99	88.18	87.05	**92.76**
Art. #2	93.18	93.28	93.45	93.52	**94.25**
Art. #3	94.28	94.40	93.70	93.55	**95.42**
Art. #4	89.32	89.32	86.90	86.43	**91.63**
Art. #5	91.59	91.88	89.87	89.90	**92.20**
Print #1	88.25	93.20	**94.56**	92.20	NA
Print #2	90.64	95.56	**95.99**	93.33	NA
Print #3	91.68	94.51	**95.71**	93.55	NA
Print #4	90.84	94.90	**95.15**	94.11	NA
Egg	92.08	98.89	**99.54**	99.13	

Table 2: Classification accuracies achieved by CART classifier when using different object extraction methods, the best performing ones underlined in bold font

5.3 Results

We present the results in three tables for the three different classifiers (CART, SVMs and poss.NN), where in each of these tables the rows represent the various data sets (five artificial ones ArtifData01-05, one for eggs and four for CD imprint data CDImprint01-04) and the columns represent the various object extraction approaches. The single values in the tables represent the 10-fold CV accuracy. The significance of the results in terms of the standard deviation among the 10 different folds is quite low for all data sets, i.e. smaller than 0.5% accuracy, which means that an increase of more than 0.5% accuracy is already significant. Table 2 shows us the results obtained when using CART classifier. One can realize that for the artificial data set all methods perform equally, except the normalized cut approach, which brings significant improvement (up to 5%) compared to the other methods. This is not a big surprise when taking into account that the mean absolute error on the number of objects achieved by this method is about half as high as for the other methods (note that the number of objects is known in this data set as artificially generated). For the CD imprint data the situation is a bit different, as DBSCAN can outperform all other clustering approaches and the conventional connected components algorithm is far behind the others. This is because usually objects contain pixels which are not connected, and hence connected components generates by far too many objects (as e.g. shown in Figures 4c and 4d). For the egg data the situation is similar (connected components falls short in classification accuracy by more than 6%), whereby DBSCAN can outperform the other clustering approaches. In fact, it is remarkable that DBSCAN can even improve the already high accuracy of 98.89% from hierarchical clustering to 99.54%, which means that the error (100%-accuracy) is reduced by more than a half.

SVMs	CC	HC	DBSCAN	RDG	NCUT
Art. #1	87.68	87.89	88.50	87.80	**91.02**
Art. #2	93.45	93.55	93.98	93.52	**94.64**
Art. #3	93.70	93.66	93.52	93.32	**94.38**
Art. #4	90.20	90.43	88.43	88.73	**93.32**
Art. #5	90.77	91.11	90.33	88.72	**93.32**
Print #1	88.74	93.59	**94.92**	93.22	NA
Print #2	93.45	96.73	**96.82**	95.05	NA
Print #3	92.03	94.90	**95.71**	94.10	NA
Print #4	95.10	95.10	**96.25**	94.77	NA
Egg	92.53	98.48	**99.60**	98.98	

Table 3: Classification accuracies achieved by SVM classifier when using different object extraction methods

poss. NN	CC	HC	DBSCAN	RDG	NCUT
Art. #1	83.08	83.22	82.80	82.00	**86.06**
Art. #2	89.10	89.31	88.52	88.38	**89.32**
Art. #3	91.38	91.12	91.88	90.95	**92.60**
Art. #4	84.86	85.17	86.30	84.37	**90.50**
Art. #5	90.02	90.31	88.88	88.13	**90.39**
Print #1	87.20	93.20	**94.56**	92.98	NA
Print #2	93.64	96.01	**95.99**	95.43	NA
Print #3	91.82	93.79	**95.71**	92.34	NA
Print #4	91.96	**95.56**	95.15	92.83	NA
Egg	91.07	98.62	**99.74**	99.39	

Table 4: Classification accuracies achieved by poss. NN classifier when using different object extraction methods

Now, the remaining question is: are these results really significant? Or the other way round, is the classification method maybe biased in a way that it favors a specific object extraction method? For answering this question we applied two other types of classifiers, SVMs and possibilistic neural networks, whose classification accuracies on the same data sets and object extraction methods are shown in Tables 3 and 4. These tables confirm the statements made above for the different data sets (and hence no favor of any object recognition approach towards a specific classification method is observed): for artificial data normalized cut performs best, while for CD imprint and egg data DBSCAN can outperform all others, leaving connected components far behind. Please note, that in all tables no evaluation with the normalized cut clustering algorithm on the CD imprint data could be carried out (hence shown as NA), due to a too heavy memory usage (virtual memory overflow) and too high computational effort (eigenvector calculations) inherent to this kind of algorithm, when a high number of pixels has

to be processed[6]. However, experiments have been carried out with contrast images, containing only a small number of bright pixels, which show that the normalized cut has the potential to improve the classification accuracy (at least) in a similar way as DBSCAN does.

6 Conclusion

In this paper, it was examined how various object extraction approaches in contrast images affect the predictive performance of the image classifiers trained from a pre-defined set of training images (based on features extracted from the objects found). This is an important aspect at the surface inspection system, as the image classifiers are responsible for the quality of the production items resp. the costs for re-production and customer complaints. The final conclusion is that clustering approaches can outperform standard methods such as connected components algorithm significantly and that different clustering algorithms may result in different accuracies of the classifiers. Basically, there is a favor for DBSCAN algorithm as it was among the top performers for all data sets and classification methods. Furthermore, this clustering approach is optimized for processing a large number of data points, which is usually needed, when grouping large clouds of pixels. Three different well-known classification methods were used in order to eliminate an eventual bias of a classifier when coupled with a certain object extraction method. In this sense, and as we used ten different data sets, the final statements about the impact of the object extraction approaches onto classifier performance can be seen as quite significant. Nevertheless, in future work also other clustering methods, especially those which have been developed for time-critical applications, i.e. grid-based clustering [31], have to be analysed. Also, there may be an improvement, when clustering the objects found by the connected component algorithm instead of the original bright pixels (which means using the connected component algorithm as a preprocessing step before clustering). Future enhancements may also include the incorporation of application-specific background knowledge in terms of clustering constraints [30] and/or the combination of different clustering results using so-called cluster ensembles [28].

[6]Some CD imprint images contain up to a few hundred thousands of bright pixels.

Acknowledgements

This work was supported by the European Commission (project Contract No. STRP016429, acronym DynaVis). This publication reflects only the authors' views.

References

[1] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Chapman and Hall, Boca Raton, 1993.

[2] P. Caleb-Solly and J.E. Smith. Adaptive surface inspection via interactive evolution. *Image and Vision Computing*, 25(7):1058–1072, 2007.

[3] M. Daszykowski, B. Walczak, and D. L. Massart. Looking for Natural Patterns in Data. Part 1: density based approach. *Chemometrics and Intelligent Laboratory Systems*, 56(2):83–92, 2001.

[4] Martin Ester, Hans-Peter Kriegel, Joerg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.

[5] G. Gan, C. Ma, and J.Wu. *Data Clustering: Theory, Algorithms and Applications*. Siam, Society for Industrial and Applied Mathematics, American Statistical Association, 2007.

[6] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing 2nd Edition*. Prentice Hall, Inc., New Jersey, USA, 2002.

[7] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2/3):107–145, 2001.

[8] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer Verlag, New York, Berlin, Heidelberg, Germany, 2001.

[9] B. Hopkins. A new method for determining the type of distribution of plant individuals. *Annals of Botany*, 18:213–226, 1954.

[10] J. Iivarinen and J. Rauhamaa. Surface inspection of web materials using the self-organizing map. In *Proc. SPIE Vol. 3522, Intelligent Robots and Computer Vision XVII: Algorithms, Techniques, and Active Vision, David P. Casasent; Ed.*, pages 96–103, 1998.

[11] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[12] A.A. Hashim J.G. Campbell and F. Murtagh. Flaw detection in woven textiles using space-dependent fourier transform. In *ISSC '97, Irish Signals and Systems Conference, F.J. Owens, editor*, pages 241–252.

[13] C.W. Kim and A.J. Koivo. Hierarchical classification of surface defects on dusty wood boards. *Pattern Recognition Letters*, 15:713–712, 1994.

[14] T. Kubota, P. Talekar, X. Ma, and T.S. Sudarshan. A nondestructive automated defect detection system for silicon carbide wafers. *Machine Vision and Applications*, 16:170–176, 2005.

[15] M. Niskanen. *A Visual Training based Approach to Surface Inspection*. PhD thesis, Department of Electrical and Information Engineering, University of Oulu, June 2003.

[16] S. Ozdemir, A. Baykut, R. Meylani, and A. Ertüzün A. Erçil. Comparative evaluation of texture analysis algorithms for defect inspection of textile products. In *Proceedings Int. Conf. on Pattern Recognition*, pages 1738–1741.

[17] G. Papari and N. Petkov. Algorithm that mimics human perceptual grouping of dot patterns. In *Proc. First Int. Symp. on Brain, Vision and Artificial Intelligence BVAI, Naples*, volume 3704, pages 497–506. Springer-Verlag Berlin Heidelberg, October 2005.

[18] F. Pernkopf. 3d surface analysis using coupled hmms. *Machine Vision and Applications*, 16(5):298–305, 2005.

[19] Thomas A. Runkler. *Information Mining: Methoden, Algorithmen und Anwendungen intelligenter Datenanalyse*. Vieweg, Gabler - Computational Intelligence, April 2000.

[20] S. Salvador and P. Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *ICTAI 04*, 2004.

[21] M. Schael. Texture fault detection using invariant textural features. In B.Radig and S. Florczyk, editors, *Proceedings of DAGM 2001, Pattern Recognition*, volume 2191 of *LNCS*, pages 17–24. Springer, Berlin Heidelberg, 2001.

[22] B. Schölkopf and A.J. Smola. *Learning with Kernels - Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, London, England, 2002.

[23] Linda G. Shapiro and George C. Stockman. *Computer Vision*. Prentice Hall, Inc., New Jersey, USA, 2001.

[24] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[25] R. Sibson. Slink: An optimally efficient algorithm for the single-link cluster method. *Computer Journal*, 16(1):30–34, 1973.

[26] M.L. Smith. *Surface Inspection Techniques: Using the Integration of Innovative Machine Vision and Modelling Techniques*. Professional Engineering Publishing, 2000.

[27] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, 36:111–147, 1974.

[28] Alexander Strehl and Joydeep Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, December 2002.

[29] V. Vapnik. *Statistical Learning Theory*. Wiley and Sons, New York, 1998.

[30] K. Wagstaff and C. Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–1110, 2000.

[31] Wei Wang, Jiong Yang, and Richard R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Twenty-Third International Conference on Very Large Data Bases*, pages 186–195, Athens, Greece, 1997. Morgan Kaufmann.

[32] P.D. Wasserman. *Advanced Methods in Neural Computing*. Van Nostrand Reinhold, New York, 1993.

[33] D.H. Wolpert. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.