Customized error clustering of industrial surface inspection images

Melinda Pap
pap.melinda@uni-eszterhazy.hu

Eszterházy Károly University
Eger, Hungary

Abstract

In industrial quality control the optical surface inspection of the produced items frequently plays an important role. The typical aim of industrial visual inspection is to determine whether the produced item meets certain requirements or contains errors. In our particular dataset, due to the nature of the errors and the special properties of the construction of the surface inspection images, the error-regions in most cases are not connected any more.

The goal of our study is to provide a clustering process that finds reasonable grouping of the disconnected error regions, such that the inspection of these images can be automated. By leaving out the human supervision, we can reduce the uncertainty and inconsistency, and accelerate the inspection. Furthermore, our aim was to find a computationally efficient method that does not require high level domain knowledge and is able to extract the required information only from the analysed images themselves. We set up a clustering pipeline, select the proper methods for preprocessing, experiment with several available clustering methods (SLINK [FLP+51], DBSCAN [EpKSX96], TURN* [FZ02],RDG [PP05] and CTS [FJ02]) and observe the possibilities of the automation of parameter selection. We evaluate the clustering methods with the aid of external validity indices, but we also propose an internal validity index for the case when no ground truth is available [HBV01].

1 Introduction

In industry, inspection is the process of determining if a product deviates from a given set of specifications. It is used, for instance, to determine if the quality of the product is acceptable for further usage, to guide the process stages, to collect statistical information, and to sort the final products according to their quality or appearance. Examples of industrial visual inspection applications can be found in [MTPL13, BKJJ13]. This task can be divided into several steps illustrated in Fig. 1, where each step is checking a specific criterion [DSAW+99].

In conventional industrial surface inspection processes, the image of the observed product has to undergo some preprocessing steps before it is analysed by image processing tools. In most cases it is known which image

In: E. Vatai (ed.): Proceedings of the 11th Joint Conference on Mathematics and Computer Science, Eger, Hungary, 20th – 22nd of May, 2016, published at http://ceur-ws.org

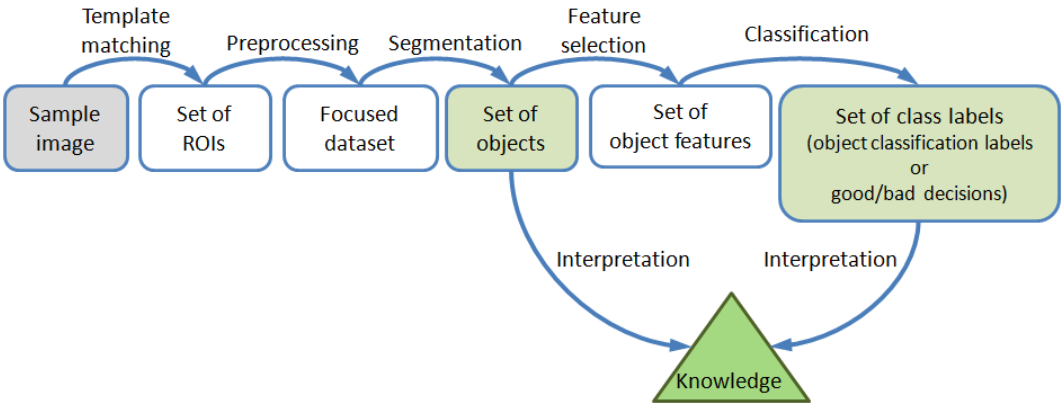


Figure 1: Steps of a typical industrial visual inspection task

areas are needed to be analysed, thus they can be isolated from the rest of the image. This restriction results in image information of reduced size that accelerates the further processing. A specific isolation process of objects is the extraction of regions of interests (ROIs).

In some cases an ideal template of the product is stored for verification. This template is not necessarily a real image taken from a physically existing "master piece", but can be a computer generated one. If such information is available it simplifies the task of ROI extraction to a simple image arithmetic operation i.e. template matching or image subtraction [NJ95]. The next step is to preprocess our dataset, this frequently means noise removal and dimensionality reduction. The next stage is to segment the images to gain the objects present in them. Depending on our task, we either stop here, if our goal was only to find objects, or continue to support a classification task by feature selection. The latter is the most frequently used when it comes to quality assurance, where we want to sort and differentiate the products or decide whether to accept or reject them.

It is important to note that digital image processing in many applications is still dependent on the human experts. A human perceives most of the information by her/his sense of vision and visual inspection has undoubtedly been used in industry since the very beginning, long before computers. There are still tasks that yet can only be solved by humans effectively. One example of such case is the tasks of perceptual grouping. The human visual system can detect many classes of patterns and statistically significant arrangements of image elements.

However image processing tasks performed by human operators are more sensitive to errors, more subjective and dependent on the experience of the operator. Furthermore, automated inspection allows objects to be inspected in environments unsafe for people or where the production is done with high speed (no need to stop the product line and position the piece for inspection). Moreover, automated inspection often results in lower costs and improved and more consistent quality [NJ95].

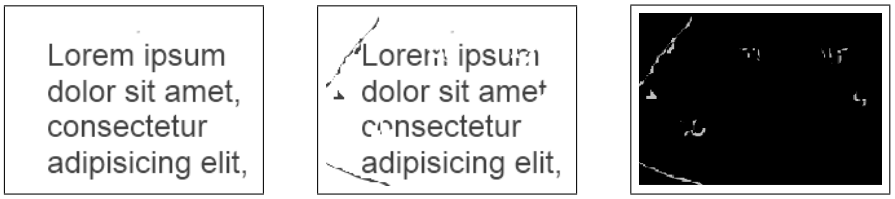
Several paper has been published in the field of industrial image processing. Some of them aim at fault detection using a template image for the ROI extraction [BKJJ13, CP16]. However only a few deals with further clustering or segmentation of the extracted ROIs [EHL$^+$10, LGW$^+$17].

The rest of this paper is organized as follows. First the problem is introduced in detail, then our solution proposal is explained, which involves the pre-processing and clustering steps. Afterwards we provide a parameter selection method that is one of the contributions of this paper. The results and discussion are presented in Section 4., where multiple clustering methods are evaluated. Finally, conclusions and further work are out-lined in Section 4.4.

2 Problem statement

Deviation images (difference images) are constructed based on a stored template: master image (reference image, template) (see Fig. 2(a)) that represents the ideal look of the type of the item that is inspected and a sample image (current image) that is the image of the currently observed item that we would like to analyse (see Fig. 2(b)). The deviation image is constructed by the subtraction of the master and the sample image, thus contains regions where the sample image deviates from the master image (see Fig.2(c)). The deviation image construction is a widely used method in print quality inspection and PCB defect detection [DSAW$^+$99, ANDAM11]. In the

case of gray-scale images, the result of the deviation image construction is a gray-scale image as well, where black pixels are representing the correct areas and bright pixels the regions where the master and the sample image deviated from each other. Hence, the bright (non black) areas are representing the regions of interest (ROIs), denoting potential error regions. Since all deviating areas are detected, the bright pixels can represent either real defects or pseudo-errors, i.e. artefacts caused by varying illumination conditions or improper image alignment.



(a) Master image (b) Sample image (c) Deviation image

Figure 2: Example of deviation image construction.

The set of the used images in this paper is coming from an industrial product line. It consists of 660 high resolution: 12264×9406 ($\approx 3m$) gray-scale deviation images of the produced items. The sample images were taken at the final stage of the production, with cameras placed above the conveyor belt, then the deviation images were constructed immediately. The produced items in this particular case are transparent disks with black, bar-code-like pattern painted on them. This pattern is circular and consists of bars of varying lengths and widths. An example can be found in Fig. 3.

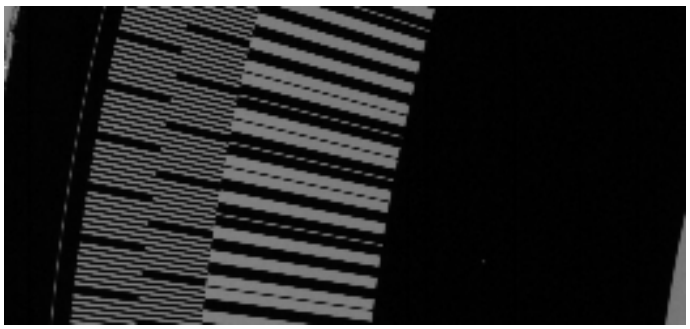


Figure 3: Slice of a sample image on which the bar-code-like pattern can be observed.

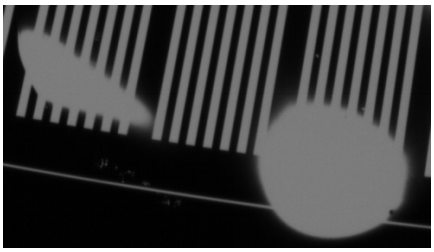
It can happen that after constructing the deviation images, the contained ROIs are not continuous any more. There can be several reasons behind this phenomenon e.g.: the improper alignment of the master and the sample image during the deviation image construction, inaccurate threshold selection, or in our particular case: the uncommon pattern, texture or colour of the master image. Examples of this phenomenon can be seen in Fig. 4.

In this scenario the company wants to know the real size of the defected areas, such that they can base their decision upon this information, e.g. reject the disk if the size of the defected area exceeds a certain threshold. We can see that in most cases the real size of the defects is greater than what appears on the deviation images. Thus, our aim is to find which objects belong to one error region and by that support such decision making.

3 Our approach

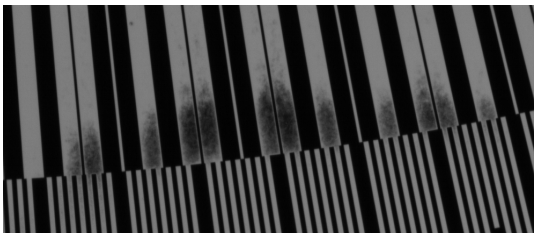
Our aim is to perform the grouping of the non-continuous bright pixel segments on the deviation images without any knowledge about the master or the sample images. This would lead us to a more flexible and more widely applicable solution that can be used in other applications with different types of items and data as well.

The first step is to apply preprocessing on the dataset such that removing noise and reducing the dimensionality of the raw data, select relevant features that are descriptive enough to represent our dataset. The next step is to find the best grouping method. In order to achieve this, we examined several clustering algorithm that could perform the grouping on our data. The clustering algorithms that suit our needs best will be evaluated on our dataset. For this purpose we analysed the available clustering evaluation techniques as well and performed the evaluation of the results of the selected clustering algorithms.

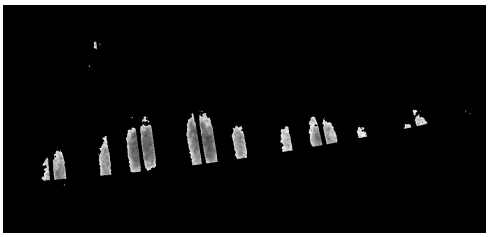


(a) Pores

(b) Pores on the deviation image



(c) Remained paint



(d) Remained paint on the deviation image

Figure 4: Example of errors and their appearance on the deviation images.

Our task here does not include the actual classification of ROIs, we do not need to conclude to which error type one ROI corresponds. However, the correct clustering is necessary to maintain information (e.g. the characterization of clusters = objects by features such as area and perimeter) that can be used for a classification process on a later stage.

3.1 Ground truth construction

We extracted 60 images from the complete dataset (cut out regions from deviation images that contain ROIs of all kinds) that will be used as "training" data. Due to the high resolution, the number of bright pixels in these images varies in the range of 500 to 600000. This subset of the complete dataset will be used to derive some conclusions about the type of ROIs and cluster structures. Moreover, the ground truth dataset is not only good for evaluation, but can be used to analyse the expected cluster structures. Table 3.1 presents some general information about the training data of which the ground truth is available.

From the analysis of the ground truth files we can conclude that we are looking for clusters of arbitrary shapes and varying sizes. Furthermore, in our project we expect clustering algorithms to assume that the pixels that belong to the same object of an image (connected component) should belong to the same cluster as well.

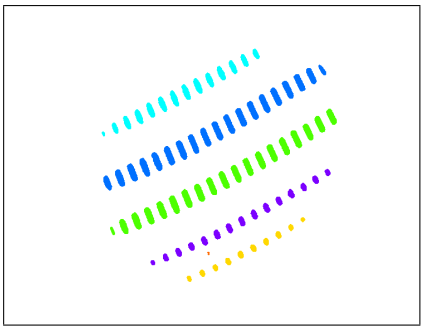


Figure 5: Examples of ground truth labelled matrices in image form. The different colours are representing the different labels.

3.2 Pre-processing

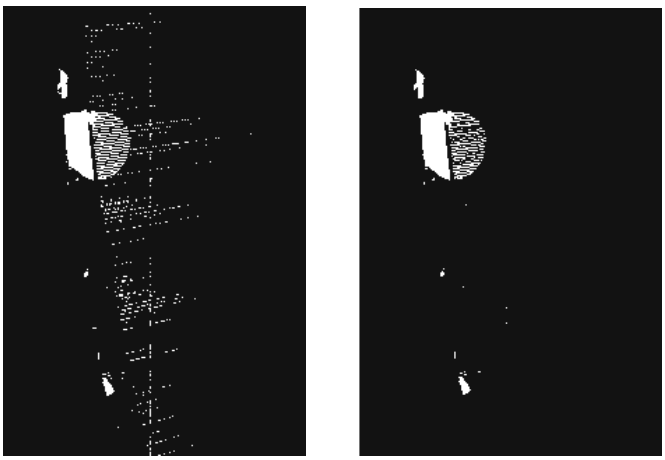
First of all, we decided to binarize the gray-scale images. This will reduce the computational costs, since we will operate on 2-dimensional logical matrices. The acceleration of clustering is highly important when it has to be

	Possible range per image [Min, Max]	
Number of points	551	540008
Number of clusters	3	112
Average cluster sizes	55.1	50724.3
Length of gap within a cluster	1	110

Table 1: General information about the training data derived from the ground truth files. The possible ranges of the number of points, the number of clusters and the average cluster sizes per image in pixels. In the last row the range of the length of the possible discontinuity within a cluster is shown i.e. the shortest distance between two objects that still belong to the same cluster.

applied during the production process. Furthermore, since our goal does not include the classification of errors, the gray-level information does not mean significantly more information.

As already mentioned before, it is often the case that images contain noise. In our case we also have to deal with the problem of pseudo errors. After the observations of the deviation images, we concluded that normally the noise and the pseudo error type of Moir patterns appear on the images as small bright areas. We decided to remove them by applying morphological filtering (morphological opening followed by closing). This will remove objects from the image that are smaller than the structuring element (in our case: a 2×2 matrix), while distorting the other objects as little as possible. The result of this operation can be seen in Fig. 6.



(a) Binarized deviation image containing pseudo errors.

(b) Binarized deviation image after applying morphological filtering.

Figure 6: Morphological filtering for pseudo error removal.

3.3 Clustering algorithm selection

When we talk about grouping process in the context of machine vision, we refer to the task of clustering. Clustering is the task of grouping a set of objects (image pixels) in such a way that objects in the same group (called cluster) are more similar (with respect to some characteristic) to each other than to those in other groups (clusters) [GMW07, TSK05]. Thus, its principle is similar to the perceptual grouping of humans.

Cluster analysis groups data points based only on the information found in the data that describes the objects and their relationships. It also helps to discover the distribution of patterns and the interesting correlations in large data sets [JMF99]. Cluster analysis is sometimes also referred to as unsupervised classification, as it creates a labelling of data points with class (cluster) labels. It is unsupervised, since there are no predefined classes and no examples that would show what kind of desirable mapping between features and classes should be valid among the data.

There are thousands of clustering algorithms available because they mainly differ in their notion of a cluster. We defined that we are looking for clusters with arbitrary shapes and varying sizes. We can assume that we need algorithms that do not require the number of clusters as an input parameter and can handle large datasets in short time, that is essential if a method is used during industrial production. In Table 2 we compared several clustering algorithms in order to see which of them suits our afore mentioned needs.

Algorithm	**Type**	**Discovers clusters with**			**Approx. time complexity**
		arbitrary shapes	**non-uniform density**	**noise**	
k-means [Mac67]	Prototype based	No	No	No	$O(n)$
CLARANS [NHS05]	Prototype based	No	No	No	$\geq O(kn^2)$
2-MSTClus [ZMW10]	Graph based	Yes	Yes	No	$O(n^2)$
RDG [PP05]	Graph based	Yes	Yes	Yes	$O(n^2)$
AUTOCLUST [ECL00]	Graph based	Yes	Partially	Partially	$O(n \log n)$
TRICLUST [LNS08]	Graph based	Yes	Yes	Yes	$O(n \log n)$
SLINK [FŁP$^+$51]	Hierarchical	Yes	No	No	$O(n^2)$
BIRCH [ZRL96]	Hierarchical	No	No	Yes	$O(n)$
CURE [GRS98]	Hierarchical	No	No	Yes	$O(n^2 \log n)$
Chameleon [KHK99]	Hierarchical	Yes	Partially	No	$> O(n \log n)$
DBSCAN [EpKSX96]	Density based	Yes	No	Yes	$O(n \log n)$
TURN* [FZ02]	Density based	Yes	No	Yes	$O(n \log n)$
ADACLUS [NLS08]	Density based	Yes	Yes	Partially	$O(n)$
FCM [Bez81]	Fuzzy prototype	No	No	Yes	$O(n)$
DifFUZZY [CMS$^+$10]	Fuzzy graph	Yes	No	No	$\geq O(n^2)$
WaveCluster [SCZ98]	Grid based	Yes	No	Yes	$O(n)$
DENCLUE 2.0 [HG07]	Grid based	Yes	No	Yes	$O(n \log n)$
BMCA [PZLB93]	Morphological	Yes	No	Yes	$O(n^2)$
Genetik k-means [PS12]	Search based	No	No	No	$O(n)$
SVC [NS06]	Search based	Yes	No	Yes	$< O(n^2)$

Table 2: Comparison of some clustering algorithms, where n is the number of data points in the dataset and k is the number of clusters.

We selected some of the available clustering algorithms for our experiments based on their complexity, sensitivity to their input parameters and whether they are capable of discovering clusters with arbitrary shapes and sizes. Hence, the following clustering algorithms were selected:

1. CC: The connected components algorithm with 8-connectivity neighbourhood, that can extract the continuous pixel regions. This method is already used in our specific industrial production line (however, with insufficient performance), thus we use its results as the baseline in our experiments.

2. CCM: The connected components algorithm extended by morphological preprocessing. This is a morphology-based clustering method that operates by applying morphological closing with the use of a disc shaped structuring element on the image before extracting the connected components in it. This is a simplified version of the BMCA [PZLB93] algorithm since in this case the dataset is already 2-dimensional and binary. This method has one parameter r, the radius of the structuring element.

3. RDG [PP05]: The Reduced Delaunay Graph algorithm is a graph-based clustering approach that constructs a Delaunay graph of the point set and cuts the edges if they exceed a certain threshold t_{RDG}. This threshold can be found by sorting the edges of the Delaunay graph according to their weights in descending order and plotting this sorted list against its rank. The knee-point of this graph will correspond to the most appropriate value for t_{RDG} as proposed in [EHL$^+$10].

4. SLINK [FŁP$^+$51]: The Single-link hierarchical clustering method that constructs a clusterring hierarchy. It starts with each point considered as a cluster and in each step the two closest clusters will be merged into a new one. The algorithm stops when all points are merged into one cluster. We used the implementation provided by the authors of [EHL$^+$10] that only calculates the cluster hierarchy until the similarity of the currently merged clusters is above a threshold called $cutOff$.

5. DBSCAN [EpKSX96]: A density-based clustering method that is based on the assumption that the clusters are dense regions in the data space, separated by regions of lower density. Each cluster is a maximum set of density-connected points. Points not connected are considered outliers. It has two parameters: the Eps that determines a radius around a point p and the second is the $MinPts$ that is the minimum number of points required within the Eps radius of the point p in order to call it an inter-cluster point. The authors of [EpKSX96] proposed an algorithm that helps the parameter selection. It works in such a way that if we set the $MinPts$ to a fixed value we can find the best value for the Eps automatically with the help of a k-distance graph.

6. TURN* [FZ02]: A density-based clustering method that operates on multiple resolutions of the data, that requires no input parameters. It consists of an overall algorithm called TURN and two component algorithms: TURN-RES and TurnCut. The first is an efficient resolution dependent clustering algorithm which returns both a clustering result and global statistics corresponding to that result. The latter is an automatic method for finding the optimum resolutions from a set of resolution results from TURN-RES. TurnCut detects a change (knee-point) in the third differential of the series of the returned statistics across resolution built by repeated calls to TURN-RES by the TURN* algorithm.

7. CTS [FJ02]: The Connected-triple based similarity method is a link-based consensus clustering algorithm that analyses the co-occurrence of points in the same cluster among the partitions of the cluster ensemble. It uses single-link clustering in its consensus function. The CTS has three parameters: the number of partitions in the ensemble m, the decay factor DC and a threshold t that is used in the consensus function. According to the authors of [IoG10], a suitable value of m is in the range $[10, 30]$ and setting the value of DC to $0,8$ usually leads to good results.

3.4 Resampling

It comes from the nature of our data that we prefer clustering algorithms that group two pixels into one cluster if they belong to the same object (connected component) of an image. Hence, the calculation of distances within points that are contained in the same object is not always necessary e.g. in the case of graph based clustering methods we only wish to obtain the edges between objects and not within.

Thus, we can apply some preprocessing steps in order to reduce the amount of data points to be processed. First we obtain the one pixel wide boundaries of the objects, that is achieved by applying morphological boundary extraction. Then, we extract the set of connected components $D_{con} = \{C_1, C_2, \ldots, C_m\}$ of the resulted image, where m is the number of objects. In this case the set will contain the m connected boundaries of the original objects i.e. the object representatives. After obtaining the set of object representatives D_{con}, the graphs used by clustering algorithms are created by analysing the distances between the pairs of object representatives C_i and C_j.

We have already mentioned the consensus clustering methods proposed in the literature. The core idea of such methods is to aggregate several input data clusterings to generate a single output clustering, thus the first step of a consensus clustering method is to generate a cluster ensemble that contains several partitions of the original dataset.

Most of the consensus clustering methods that use point co-occurrence as their consensus function, only require the set of cluster labels associated to each point in the dataset and do not need the coordinate of the points to generate the output clustering. If we use base clustering algorithms that do not violate our definition of the expected cluster structure, then we can expect that two points that belong to the same connected component of an image will belong to one cluster in any of the partitions of the cluster ensemble. In this case, we can conclude that we do not need to consider the co-occurrence of two points that belong to the same object, because those will always belong to the same cluster. Hence, we can significantly reduce the number of points used in the consensus function, by not using all the points of a connected component, but only one.

3.5 Parameter selection

A general approach that can be used with any type of clustering algorithm to find the best partitioning of the dataset automatically is based on the use of an evaluation metric e.g. any internal or external cluster validity index. This approach requires multiple runs of the specific clustering algorithm with varying parameter settings. Then the resulted partitions are rated based on a specific evaluation metric and the one that has the best rating will be returned as the final result [JD88, HBV01]. However, in most cases the parameters of clustering algorithms can vary in a wide range, thus it might require to run one clustering method hundreds or thousands of times that makes this approach extremely inefficient, especially if even the cluster validation algorithm has high complexity.

Here, we would like to propose an alternative method to find the best parameters of some clustering algorithms, without requiring them to run multiple times. Several clustering algorithms e.g. graph theoretical clustering and hierarchical clustering algorithms are based on the assumption that points within a cluster are closer to each other than to any other point not included in that cluster. The distance is the measure of dissimilarity. This statement usually holds for connectivity-based clusters with even density. These algorithms search for the clusters iteratively until a stopping criterion is fulfilled. One such stopping criterion can be a *distance threshold*, such that if the distance between two points exceeds this threshold they cannot be merged into one cluster. Other algorithms like the morphological clustering can use the distance information to size the used structuring element. The diameter of the structuring element determines the shortest distance between two clusters that can be merged into one new cluster (although the merge is also dependent on the shape of the structuring element). Such distance threshold can be the parameter of the previously mentioned DBSCAN algorithm as well, it can determine the value of the parameter *Eps*.

We decided to search for such a distance threshold with the help of a relative neighbourhood graph (RNG). This can be calculated in $O(n \log n)$ time. The steps of obtaining a proper distance threshold based on the RNG of images with continuous pixel regions are the following (illustrated in Fig. 7), where m is the number of objects (connected components):

1. Apply re-sampling by morphological boundary extraction, then extract the connected components. This results in a set that contains the m object representatives of the original objects $D_{con} = \{C_1, C_2, \ldots, C_m\}$.

2. Calculate the $m \times m$ proximity matrix P_m in which the elements $P_m(i, j)$ correspond to the shortest distances between object representatives C_i and C_j.

3. Calculate the RNG from the proximity matrix P_m.

4. Sort the edge lengths and use the L-method introduced in [SC04] to calculate the knee-point of the resulted curve. Return the found knee-point as the distance threshold.

We used this distance threshold as parameter for the CCM, SLINK and DBSCAN clustering algorithms in our experiments.

4 Evaluation

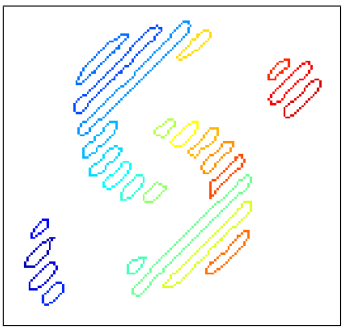
In this section we compare the results of the clustering algorithms selected previously: CC, CCM, SLINK, RDG, DBSCAN, TURN* and CTS.

We will evaluate the results of the clustering algorithms with the help of the available ground truth files mentioned before. We will determine the similarity between the ground truth and the clustering result by using the Jaccard Coefficient ($J \uparrow$) [HBV01] external validity index that can be considered as a percentage of similarity. It is calculated as follows:

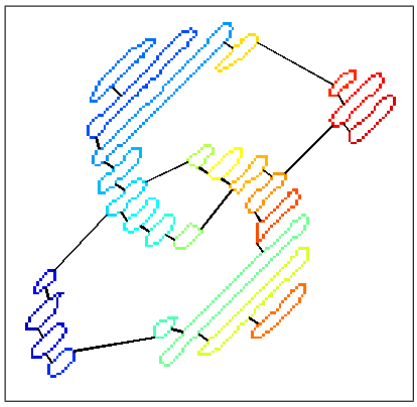
$$J \uparrow = \frac{a}{a + b + c} \tag{1}$$

where a is the number of point pairs that belong to the same cluster in the ground truth as well as in the clustering result, b is the number of point pairs that belong to the same cluster in the ground truth, but to different ones in the clustering result and c is the number of point pairs that are in different clusters in the ground truth, but in the same cluster in the clustering result.

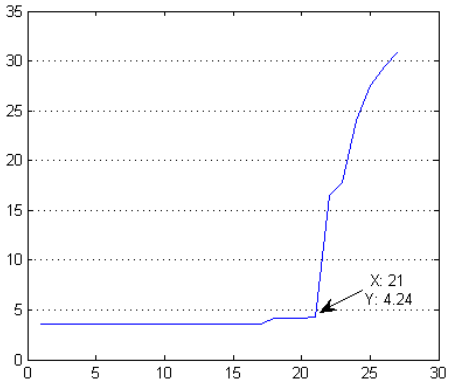
(a) Original binary image



(b) Object representatives marked by different colours



(c) RNG of the object representatives



(d) Sorted RNG edge list (a suitable knee-point is at 4.24)

Figure 7: The steps of obtaining the RNG graph of an image with continuous pixel regions. By the analysis of the list of edges, a suitable threshold to cut the graph and gain the four clusters is at 4.24.

4.1 Experimental setup

In our experiments we used a computer with 64-bit Windows Server 2012 operating system, Intel Duo CPU 2.80 GHz and 31 GB Memory.

The clustering algorithms: CCM, SLINK, DBSCAN, RDG are implemented in Matlab and were provided by the authors of [EHL+10]. We implemented the TURN* algorithm in C#. Furthermore, we used the built-in CC algorithm of Matlab. The Matlab package (LinkCluE) containing the consensus clustering method CTS was provided by the authors of [IOBG08].

4.2 Results of clustering

First of all, we compared how well the automated parameter selection method described in Sec. 3.5 performed compared to the evaluation metric based iterative approach.

We ran the three chosen algorithms (CCM, SLINK, DBSCAN) with varying parameters and evaluated them by the afore mentioned external validity index $J \uparrow$. In the case of CCM we searched for the best value for r in the range of $[5, 110]$ with step size 5. We used the same range and step size for the parameter $cutOff$ of the SLINK algorithm and for the parameter Eps of the DBSCAN algorithm. The parameter $MinPts$ of the latter algorithm was searched for in the range $[0, 5 * Eps, 12 * Eps]$.

In Table 3 we compared the results of the external evaluation metric based iterative approach and the automated parameter selection supported clustering. According to the Wilcoxon signed-rank test [Wil45] there is no significant difference between the results of one clustering algorithm using iterative and automatic parameter selection ($p > 0.05$ in all the three cases).

We compared the results of the other clustering algorithms as well. The TURN* algorithm does not require any input parameters and we used the RDG with its built in parameter selection method. In the case of the CTS method we used $DC = 0,8$ as it was proposed by the authors of the algorithm and we set the number of

Clustering method	Iterative approach	Automated parameter selection
CCM	$91.23 \pm 16.57\%$	$87.34 \pm 19.50\%$
SLINK	$93.21 \pm 13.40\%$	$89.19 \pm 18.06\%$
DBSCAN	$93.68 \pm 12.86\%$	$88.37 \pm 19.52\%$

Table 3: The average quality of clustering with standard deviation. First, the best results of iterative parameter selection, then of automatic parameter selection (described in Sec. 3.5).

ensembles m to 10. We used 5 SLINK and 5 CCM methods for the cluster ensemble generation and selected the corresponding input parameters by taking m equidistant points from the range $[5, 60]$. In this comparison we used the algorithms CCM, SLINK and DBSCAN with the automated parameter selection that we will indicate by a lower index A: CCM_A, $SLINK_A$ and $DBSCAN_A$.

Clutering algorithm	$J \uparrow$
CC	$42.49 \pm 31.75\%$
CCM_A	$87.34 \pm 19.50\%$
RDG	$82.59 \pm 23.42\%$
$SLINK_A$	$\mathbf{89.19 \pm 18.06\%}$
$DBSCAN_A$	$88.37 \pm 19.52\%$
TURN*	$83.74 \pm 21.36\%$
CTS	$85.54 \pm 21.21\%$

Table 4: Final results of the clustering algorithms. The average quality and standard deviation of the results are shown and the best result is given in bold font. The lower index A indicates that the clustering algorithm was supported by our automated parameter selection method.

In Table 4. the average results of the clustering algorithms are shown with standard deviation. We can see that the $SLINK_A$ achieved the highest accuracy with 89.19%. However, according to the Wilcoxon signed-rank test there is no significant difference between the results of the CCM_A, $DBSCAN_A$ and $SLINK_A$ algorithms ($p > 0.05$ in all the three cases).

The corresponding runtimes of the clustering algorithms are shown in Table 5. According to the Wilcoxon signed-rank test [Wil45], there is significant difference between the runtimes of the algorithms. We can see that even though the $DBSCAN_A$ performed with high accuracy on the dataset, it is almost the worst in this aspect. On the other hand, the CCM_A and $SLINK_A$ algorithms run similarly fast but they still have significant difference ($p = 0.013$) in runtime. The average runtime of the CCM_A algorithm for the complete dataset is $00 : 01 : 43$ sec, while it is $00 : 02 : 41.84$ sec for the $SLINK_A$ algorithm. Hence, we can conclude that the CCM_A algorithm is the best in the aspect of runtimes.

Now we see that the CCM_A method is the best method to choose, since it is not significantly worse than the algorithm with the highest accuracy ($SLINK_A$), but it is significantly faster.

4.3 Internal evaluation

In our experiments we used external cluster validity indices to evaluate the clustering results. Unfortunately these require the existence of ground truth files. Our motivation was to find an internal validity index that approximates well the actual accuracy of the clustering result, even if there is no ground truth data available any more. Since we know that our clusters can appear in different shapes and sizes, we focused mainly on the indices that are capable of evaluating such a dataset. We found that the graph-based internal validity indices described in [Dun73, Ilc12, PB97] were unable to handle datasets of this size unlike the indices Silhouettes ($Sil \uparrow$) [Rou87] and Davies-Bouldin ($DB \downarrow$) [DB79].

The $Sil \uparrow$ value varies in the range $[-1, 1]$, the greater the index the better. We adjusted the indices to the $[0, 1]$ scale to be able to compare it to the used external validity index $J \uparrow$. For the $DB \downarrow$ we know that the

Algorithm	CPU runtime with pixel number: 551	17593	540008
CC	00:00:00.003	00:00:00.138	00:00:00.183
CCM_A	00:00:19.95	00:09:10.14	00:22:25.29
RDG	00:00:00.91	00:14:52.02	-
$SLINK_A$	00:00:00.13	00:00:27.50	00:52:24.50
$DBSCAN_A$	00:00:01.82	05:24:56.97	18:02:02.00
TURN*	00:00:00.16	00:01:45.86	24:22:03.00
CTS	00:00:45.94	00:48:03.41	00:50:12.13

Table 5: The elapsed CPU times of clustering algorithms run on datasets of different sizes (the sizes are given in pixels). The blank cell indicates that the algorithm was not able to handle the corresponding dataset. The TURN* algorithm explored 6 resolutions. The lower index A indicates that the clustering algorithm was supported by our automated parameter selection method.

lower the index, the better the clustering result, hence, we negated the values. We also know that although the upper-bound of this index is known to be 0 in this case, there is no fix lower-bound. Thus we took the minimum value of this index on this dataset as the lower-bound, then adjusted it to the $[0, 1]$ scale. The average deviation of the $Sil \uparrow$ index to the $J \uparrow$ was 25%, while that of the $DB \downarrow$ was 14%.

5 Conclusions

The goal of our project was to provide a clustering method that finds reasonable grouping of the disconnected error regions in our set of industrial images, such that the inspection of these images can be automated. Furthermore, our aim was to find a method that does not require high-level domain knowledge and is able to extract the required information only from the analysed images themselves. Since we would like to support the industry with the found clustering method, another important criterion was that the clustering method has to be computationally efficient, such that it can be applied for product lines in real-time.

We managed to achieve these goals with 87.34% clustering accuracy by the CCM clustering method combined with our automatic parameter selection method CCM_A. Although the $SLINK_A$ method achieved 89.19% accuracy on our training data, there is no significant difference between the results of the CCM_A and the $SLINK_A$ methods, however the first is significantly faster than the latter.

With our automatic parameter selection method we avoided to run the clustering methods multiple times on one image to find the best parameters, therefore, the computation time did not increase significantly in any of the cases. Furthermore, by this method we approximated well the best parameter values, such that the results came close to the real optimal solutions (without using any external information on the true cluster structure).

We used two types of cluster validity measures, such as external and internal cluster validity indices. The first one was used to compare how well the results of the applied clustering algorithms approximated the true cluster structures, by using pre-labelled images as ground truth files. This was important to find the clustering algorithm that performs the best on our dataset. However, the ground truth files are only available for a subset of the images.

The second type of validity measures are the internal validity indices, that can be used on a later stage when no ground truth file is available any more. We found the $DB \downarrow$ index to be the closest to the external index, with 14% of average deviation. This index can be used later, along with the CCM_A clustering method to give an impression of the quality of the clustering result.

6 Acknowledgements

The research was supported by the grant EFOP-3.6.1-16-2016-00001 "Complex improvement of research capacities and services at Eszterhazy Karoly University".

References

[ANDAM11] Eckhard Ammann, Ismael Navas-Delgado, and José F. Aldana-Montes. A semantic-supported knowledge development conception. In *Proceedings book of work congress on engineering. WCE2011. London, UK, 6-8 July, 2011*, volume 2. Newswood Limited, nov 2011.

[Bez81] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[BKJJ13] Csaba Benedek, Olivér Krammer, Mihály Janóczki, and László Jakab. Solder paste scooping detection by multilevel visual inspection of printed circuit boards. *Industrial Electronics, IEEE Transactions on*, 60(6):2318–2331, 2013.

[CMS+10] Ornella Cominetti, Anastasios Matzavinos, Sandhya Samarasinghe, Don Kulasiri, Sijia Liu, Philip Maini, and Radek Erban. Diffuzzy: a fuzzy clustering algorithm for complex datasets. *International Journal of Computational Intelligence in Bioinformatics and Systems Biology*, 1(4):402–417, 2010.

[CP16] Ssu-Han Chen and Der-Baau Perng. Automatic optical inspection system for ic molding surface. *Journal of Intelligent Manufacturing*, 27(5):915–926, 2016.

[DB79] David L Davies and Donald W Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):224–227, 1979.

[DSAW+99] Christian Demant, Bernd Streicher-Abel, Peter Waszkewitz, Michaela Strick, and Gary Schmidt. *Industrial image processing : visual quality control in manufacturing*. Springer, Berlin, Heidelberg, Paris, 1999.

[Dun73] Joseph C Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. 1973.

[ECL00] Vladimir Estivill-Castro and Ickjai Lee. Autoclust: Automatic clustering via boundary extraction for mining massive point-data sets. In *In Proceedings of the 5th International Conference on Geocomputation*, pages 23–25, 2000.

[EHL+10] Christian Eitzinger, Wolfgang Heidl, Edwin Lughofer, Stefan Raiser, Jim E. Smith, Muhammad Atif Tahir, Davy Sannen, and Hendrik Van Brussel. Assessment of the influence of adaptive components in trainable surface inspection systems. *Mach. Vis. Appl.*, 21(5):613–626, 2010.

[EpKSX96] Martin Ester, Hans peter Kriegel, Jörg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[FJ02] Ana LN Fred and Anil K Jain. Data clustering using evidence accumulation. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 4, pages 276–280. IEEE, 2002.

[FŁP+51] K Florek, J Łukaszewicz, J Perkal, Hugo Steinhaus, and S Zubrzycki. Sur la liaison et la division des points d'un ensemble fini. In *Colloquium Mathematicae*, volume 2, pages 282–285. Institute of Mathematics Polish Academy of Sciences, 1951.

[FZ02] Andrew Foss and Osmar R. Zaïane. A parameterless method for efficiently discovering clusters of arbitrary shape in large datasets. In *ICDM*, pages 179–186. IEEE Computer Society, 2002.

[GMW07] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering - theory, algorithms, and applications*. SIAM, 2007.

[GRS98] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. *SIGMOD Rec.*, 27(2):73–84, June 1998.

[HBV01] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17:107–145, 2001.

[HG07] Alexander Hinneburg and Hans-Henning Gabriel. Denclue 2.0: Fast clustering based on kernel density estimation. In *Proceedings of the 7th International Conference on Intelligent Data Analysis*, IDA'07, pages 70–80, Berlin, Heidelberg, 2007. Springer-Verlag.

[Ilc12] Nejc Ilc. Modified dunn's cluster validity index based on graph theory. (2), 2012.

[IOBG08] Natthakan Iam-On, Tossapon Boongoen, and Simon Garrett. Refining pairwise similarity matrix for cluster ensemble problem with cluster relations. In *Discovery Science*, pages 222–233. Springer, 2008.

[IoG10] Natthakan Iam-on and Simon Garrett. Linkclue: A matlab package for link-based cluster ensembles. *Journal of Statistical Software*, 36(9):1–36, 8 2010.

[JD88] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[JMF99] A K Jain, M N Murty, and P. J. Flynn. Data clustering: A review, 1999.

[KHK99] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar. Chameleon: A hierarchical clustering algorithm using dynamic modeling, 1999.

[LGW+17] Xiaoqing Li, Bin Gao, Wai Lok Woo, Gui Yun Tian, Xueshi Qiu, and Liangyong Gu. Quantitative surface crack evaluation based on eddy current pulsed thermography. *IEEE Sensors Journal*, 17(2):412–421, 2017.

[LNS08] Dongquan Liu, Gleb V. Nosovskiy, and Olga Sourina. Effective clustering and boundary detection algorithm based on delaunay triangulation. *Pattern Recogn. Lett.*, 29(9):1261–1273, July 2008.

[Mac67] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[MTPL13] Miha Možina, Dejan Tomaževič, Franjo Pernuš, and Boštjan Likar. Automated visual inspection of imprint quality of pharmaceutical tablets. *Machine vision and applications*, 24(1):63–73, 2013.

[NHS05] Raymond T. Ng, Jiawei Han, and Ieee Computer Society. Clarans: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, pages 1003–1017, 2005.

[NJ95] T.S. Newman and A.K. Jain. A survey of automated visual inspection. *CVIU*, 61(2):231–262, March 1995.

[NLS08] Gleb V. Nosovskiy, Dongquan Liu, and Olga Sourina. Automatic clustering and boundary detection algorithm based on adaptive influence function. *Pattern Recognition*, 41(9):2757–2776, 2008.

[NS06] J Saketha Nath and Shirish Krishnaj Shevade. An efficient clustering scheme using support vector methods. *Pattern Recognition*, 39(8):1473–1480, 2006.

[PB97] N. R. Pal and J. Biswas. Cluster validation using graph theoretic concepts. 30(6):847–857+, 1997.

[PP05] Giuseppe Papari and Nicolai Petkov. Algorithm that mimics human perceptual grouping of dot patterns. In Massimo De Gregorio, Vito Di Maio, Maria Frucci, and Carlo Musio, editors, *BVAI*, volume 3704 of *Lecture Notes in Computer Science*, pages 497–506. Springer, 2005.

[PS12] Piyaphol Phoungphol and Inthira Srivrunyoo. Boosting-genetic clustering algorithm. In *ICMLC*, pages 1218–1223, 2012.

[PZLB93] J.-G. Postaire, R. D. Zhang, and C. Lecocq-Botte. Cluster analysis by binary morphology. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(2):170–180, feb 1993.

[Rou87] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, 1987.

[SC04] Stan Salvador and Philip Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on*, pages 576–584. IEEE, 2004.

[SCZ98] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. pages 428–439, 1998.

[TSK05] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining.* Addison-Wesley, 2005.

[Wil45] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, pages 80–83, 1945.

[ZMW10] Caiming Zhong, Duoqian Miao, and Ruizhi Wang. A graph-theoretical clustering method based on two rounds of minimum spanning trees. *Pattern Recogn.*, 43(3):752–766, March 2010.

[ZRL96] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996.