s c c h
software competence center
hagenberg

FUZZY.LOGIC.LAB.LINZ

Advances in Knowledge-Based Technologies

Proceedings of the Master and PhD Seminar Summer term 2008, part 2

Softwarepark Hagenberg
SCCH, Room 0/2
June 18th, 2008

Software Competence Center Hagenberg
Softwarepark 21
A-4232 Hagenberg
Tel. +43 7236 3343 800
Fax +43 7236 3343 888
www.scch.at

Fuzzy Logic Laboratorium Linz
Softwarepark 21
A-4232 Hagenberg
Tel. +43 7236 3343 431
Fax +43 7236 3343 434
www.flll.jku.at

Program

9:00–10:30 Session 1 ***(Chair: Roland Richter)***

9:00 Stefan Raiser:
Object Extraction with Clustering Methods in Industrial Machine Vision Applications

9:30 Henrike Stephani:
Clustering Terahertz-Spectra and Text-Documents

10:00 Tomas Kazmar:
Opacity Quantification in Cardiac Angiogram Sequences

10:30 Coffee Break

10:45–12:45 Session 2 ***(Chair: Bernhard Moser)***

10:45 Stephan Winkler:
Evolutionary System Identification

11:15 Leila Muresan:
Regularization in diffusion tensor images

11:45 Szilard Pall:
GPU Programming Approach for Parallelizing Support Vector Machines

Technical Report
FLLL–TR–2008-02

FUZZY.LOGIC.LAB.LINZ

Object Extraction with Clustering Methods in Industrial Machine Vision Applications

Stefan Raiser
Fuzzy Logic Laboratorium Linz-Hagenberg
e-mail stefan.raiser@jku.at

Institut für Wissensbasierte Mathematische Systeme
Johannes Kepler Universität Linz
Altenberger Str. 69
A-4040 Linz
Austria

Fuzzy Logic Laboratorium Linz
softwarepark hagenberg
Softwarepark Hagenberg
Softwarepark 21
A-4232 Hagenberg
Austria

Abstract

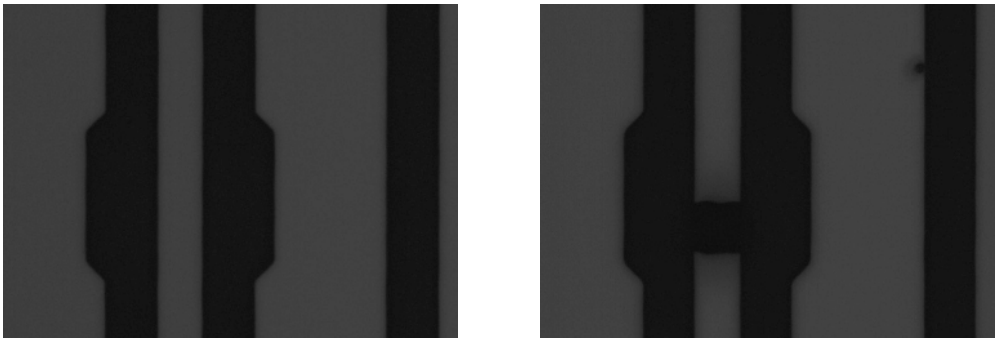
This paper investigates the applicability of clustering methods to perform object extraction from digital images as they occur in industrial vision applications like surface inspection. Especially, the ability of these methods to find non-connected objects like discontinous scratches or widespread ink splashes is examined.

The introduction describes the problem of object extraction and the commonly used algorithm for solving this task. In the following chapter, clustering methods, originally developed in the fields of statistics and data mining, are presented as an alternative approach for object extraction. In the next chapter, the previously described methods are applied on artificial as well as on real images from various surface inspection applications and the results are discussed. The last chapter gives a conclusion based on the gained experience and provides some outlook on future work.

Keywords: object extraction, clustering, cluster tendency, cluster validation

1 Introduction

One of the major steps in image processing is image segmentation, which means to extract regions from the image that correspond to the objects the user is interested in. In surface inspection, these interesting objects are all deviations from the "perfect" part, because they may represent an unacceptable imperfection or even a severe defect. Figure 1a shows the partial image of a thin-film sensor as it should look like. It therefore serves as a master image. In Figure 1b an arbitrary test image is displayed. There are two areas in the test image, which are different compared to the master and therefore should be detected during image segmentation.



(a) (b)

Figure 1: a) Master image, b) Test image

The first step now is to calculate the absolute difference between master and test image. The resulting values make up a difference image, where all non-zero pixels represent a deviation from the ideal master. Figure 2 shows an 8-bit grayscale difference image derived from the sample presented in Figure 1a and 1b. Most of the image is black, whereas the two deviating regions consist of rather bright grayvalues.

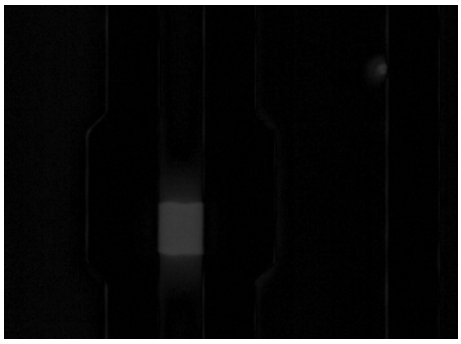


Figure 2: Difference image.

As next step, a global thresholding, which is the simplest segmentation algorithm, is applied on the difference image. The threshold operation sets all pixels, which have a grayvalue above a certain level to one and the remaining pixels to zero. So thresholding produces a binary image

with a black (0) background and a white (1) foreground, which contains the objects of interest.

When the thresholding operation was carried out successfully, the foreground of the binary image contains multiple (interesting) objects, that should be returned individually. In other words, the white pixels have to be grouped together somehow, in order to extract distinct regions. Typically, the objects of interest 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 this two definitions is called a connected component.

To obtain the individual regions, all connected components of the binary image have to be computed. In [1] 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. Figure 3 shows the result of the connected component algorithm on the sample test image.

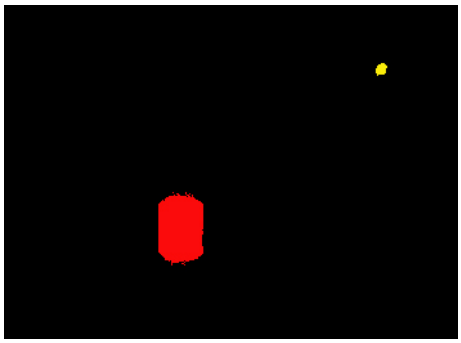


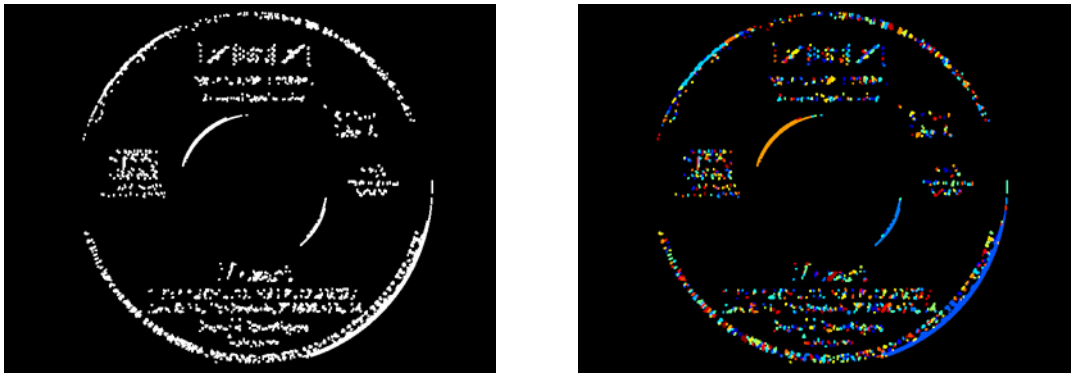
Figure 3: Labeled image with two objects found.

Till now is was assumed, that an object consists of a connected set of pixels. But in a variety of surface inspection environments, non-connected objects like discontinous scratches or widespread ink splashes can occur. Figure 4a shows an example from a CD imprint inspection. In this case the connected component algorithm finds way too many objects (like 1090 objects as in Figure 4b).

For a human it is quite easy to find the natural groupings in the thresholded image, because he or she is able to perceive patterns, even if they are very fragmented. So, the challenge is to exploit new methods, able to find correctly all objects, which consist of white pixels that visually belong together, but need not necessarily be connected.

2 Object Extraction with Clustering Methods

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"



(a) (b)

Figure 4: a) Thresholded image, b) Labeled image (1090 objects found)

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 choosen similarity/dissimilarity measure. In other words, clustering represents an unsupervised classification technique, that assigns each data point into a previously unknown category. A comprehensive description of clustering can be found i.e. in [2], [3], [4] or [5].

These characteristics make clustering a candidate for solving the object extraction task described in the previous chapter. The basic idea is to consider the white pixels in the binary difference image as data points, which serve as input for the clustering algorithm. 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 in shown in Figure 5.

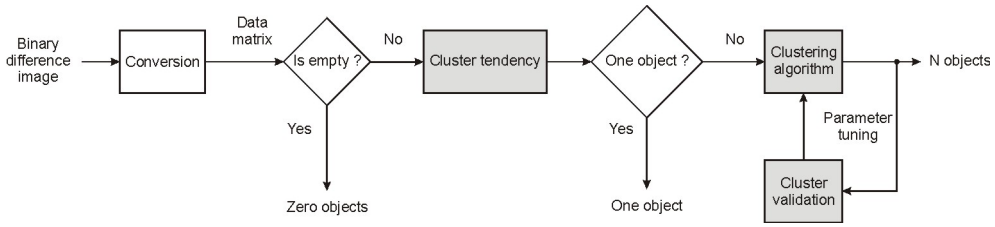


Figure 5: Clustering methods for object extraction.

After the binary difference 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, cluster validation techniques can be used to tune the parameters of the clustering algorithm.

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 white 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 (aka Hopkins test) as described by [6] and [7]. How the Hopkins index can be incorporated in the object extraction application flow is shown in Figures 6a and 6b. It can be used as a stand-alone test for cluster tendency (Figure 6a) or be combined with the connected component algorithm (Figure 6b).

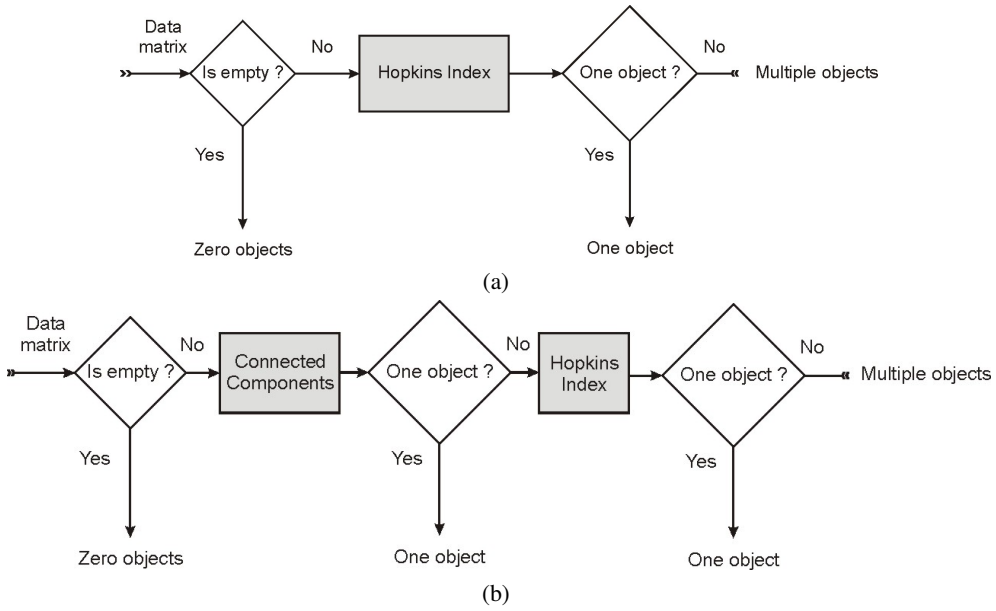


Figure 6: Verifying the existance of (multiple) objects a) one-stage approach, b) two-stage approach

Clustering algorithm

After checking the existance 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 extract the distinct objects. It's major task is to find the natural grouping of the white

pixels and to assign each pixel to a cluster (object).

For the object extraction task, a clustering algorithm is needed, which can detect clusters of arbitrary shape, because the white pixel areas in the binary image can exhibit any form. After the grouping, each white pixel should be assigned to a distinct object. In terms of clustering, 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. Depending on the cycle times of the application, the computational complexity of the algorithm has to be considered.

The following clustering algorithms are all capable of finding arbitrarily shaped objects in a binary image and their performance has been investigated for a variety of different object extraction applications, described in chapter 3:

- **Single linkage clustering** [8] represents one common variant of the hierarchical algorithms.
- **DBSCAN** [9] is a density-based algorithm, which can handle large data sets. Because of its definition of clusters as high density regions, it is very suitable in the object extraction context.
- **Recuced Delaunay graph** [10] is representing a graph-based approach.
- **Normalized cut** [11] is a spectral clustering method.

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.

Another approach is to use so-called cluster validation techniques (see [12]) to determine an appropriate parameter setting or to perform an automatic fine-tuning of estimated parameter values during on-line operation as illustrated by Figure 7a.

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.

Figure 7b illustrates how this procedure can be incorporated in the object extraction framework. 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. The clustering part can be implemented as concurrent processes as shown in the figure, if this is supported by the machine vision hardware setup.

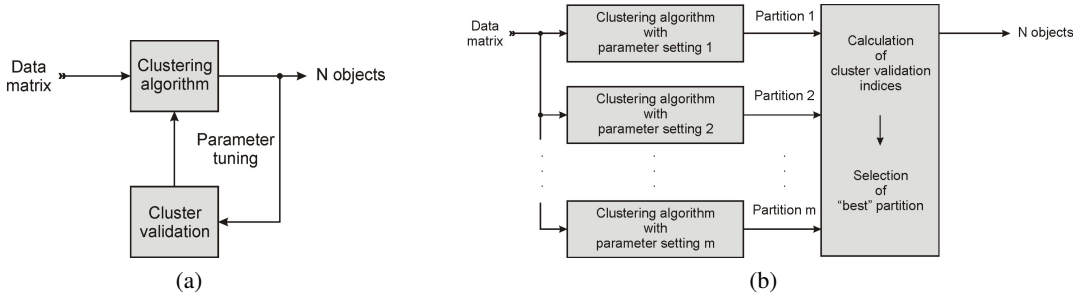


Figure 7: a) Using cluster validation for parameter tuning. b) Determining the best partition with CV indices.

It has to be kept in mind, that also the cluster validation test has to be able to handle arbitrary shapes like the clustering algorithm itself and the computational complexity for doing the cluster quality assessment is again an important issue.

3 Application Examples

This chapter discusses four application examples, on which the object extraction performance of the previously mentioned clustering methods has been tested. The first one deals with artifically created images, where the number of objects is known and can be used to evaluate the performance of the different approaches. The other examples operate on real images, derived from various surface inspection applications. Here, no expert knowledge is available about the objects actually contained in the images, so the assessment of the object extraction quality has been carried out simply by visual judgement.

3.1 Artificial Data

Overview

Five data sets, each containing 20.000 artificially created difference images, have been available for testing various object extraction methods. The images have a size of 128x128 pixels and are grayscale, hence have to be binarized by a threshold operation. Here, in order to include all deviating (non-zero) pixels, the threshold value was set to zero. The objects contained in the images mostly possess ellipsodial and compact shapes. Since the number of objects for each image is known, a quantitative evaluation of the clustering methods, based on the difference between the objects found and their real number, is possible. However, the exact object locations are not available and therefore the groupings could not be checked on a pixel-wise basis.

Evaluation

First, the performance of the Hopkins index in detecting images with single clusters was tested. In Table 1 the results are listed.

	Performance of Hopkins Index (Threshold=0.6)			
	# images	# single clusters	found [%]	overdetected [%]
ArtifData01	19224	1213	99.34	0.29
ArtifData02	18579	2468	97.93	0.66
ArtifData03	19172	2337	89.52	0.13
ArtifData04	19819	2018	83.60	1.02
ArtifData05	18211	802	81.67	0.08

Table 1

Column *#images* shows, that not all of the 20.000 images per set were used, as some of them have too less white pixels to compute the Hopkins Index. In *#single clusters* the amount of images actually containing a single object is listed. The following column shows the detection rate, which reflects the percentage of images with a single object correctly identified by using the Hopkins index. Not surprisingly, for *ArtifData01* the percentage is very high (99.34%), because the threshold value of 0.6 was choosen according to maximize the detection rate for this data set. The other data sets also perform quite well with rates above 80 %. It has to be mentioned, that even if the Hopkins Index fails in detecting an image with a single object, the subsequent clustering algorithm (at least all methods used here) is able to handle and detect a single object. In the rightmost column the overdetection rate is listed, which represents the percentage of images spuriously considered to contain one object. In this case, no more clustering (object extraction) is carried out after the calculation of the Hopkins Index and the white pixels are incorrectly grouped together. Fortunately, here the overdetection rates are very low (except for *ArtifData04*).

When evaluating the different clustering approaches, the parameters of the algorithms were set to fixed values or determined by automatic procedures.

In order to quantify the object extraction performance, the number of objects found was compared with their real number. Table 2 shows the mean absolute error (MAE) between the number of objects found and the real number of objects. Hence, it addresses the question, how the clustering over- or underestimates the number of objects on an average. In nearly all cases except for *ArtifData04* the clustering algorithms show a smaller error than CC. In particular RDG, DBSCAN and NCUT reduce the MAE significantly.

Of course, the results presented so far can only give a hint on how good an object extraction methods performs, since only object numbers are compared here. Therefore on the following pages two example images illustrate the actual grouping of the white pixels according to the different approaches[1].

[1]The colors, used to mark the objects found, have no meaning and can differ between the clustering algorithms even

	MAE between number of real and extracted objects				
	CC	HC-SL	RDG	DBSCAN	NCUT
ArtifData01	1.45	1.20	0.58	0.50	0.26
ArtifData02	2.07	1.16	0.66	0.58	0.41
ArtifData03	1.64	1.81	1.27	0.69	0.67
ArtifData04	0.93	1.59	1.17	1.06	0.92
ArtifData05	2.74	2.33	1.98	0.96	0.76
Mean	1.76	1.62	1.13	0.76	0.60

Table 2

Conclusion

In the case of the five artificial image data sets, clustering methods perform quite well and are superior to the connected component algorithm. This outcome is not surprising, since the objects contained in the images are very nice in terms of compactness and simplicity of shape.

when the objects have been extracted identically.

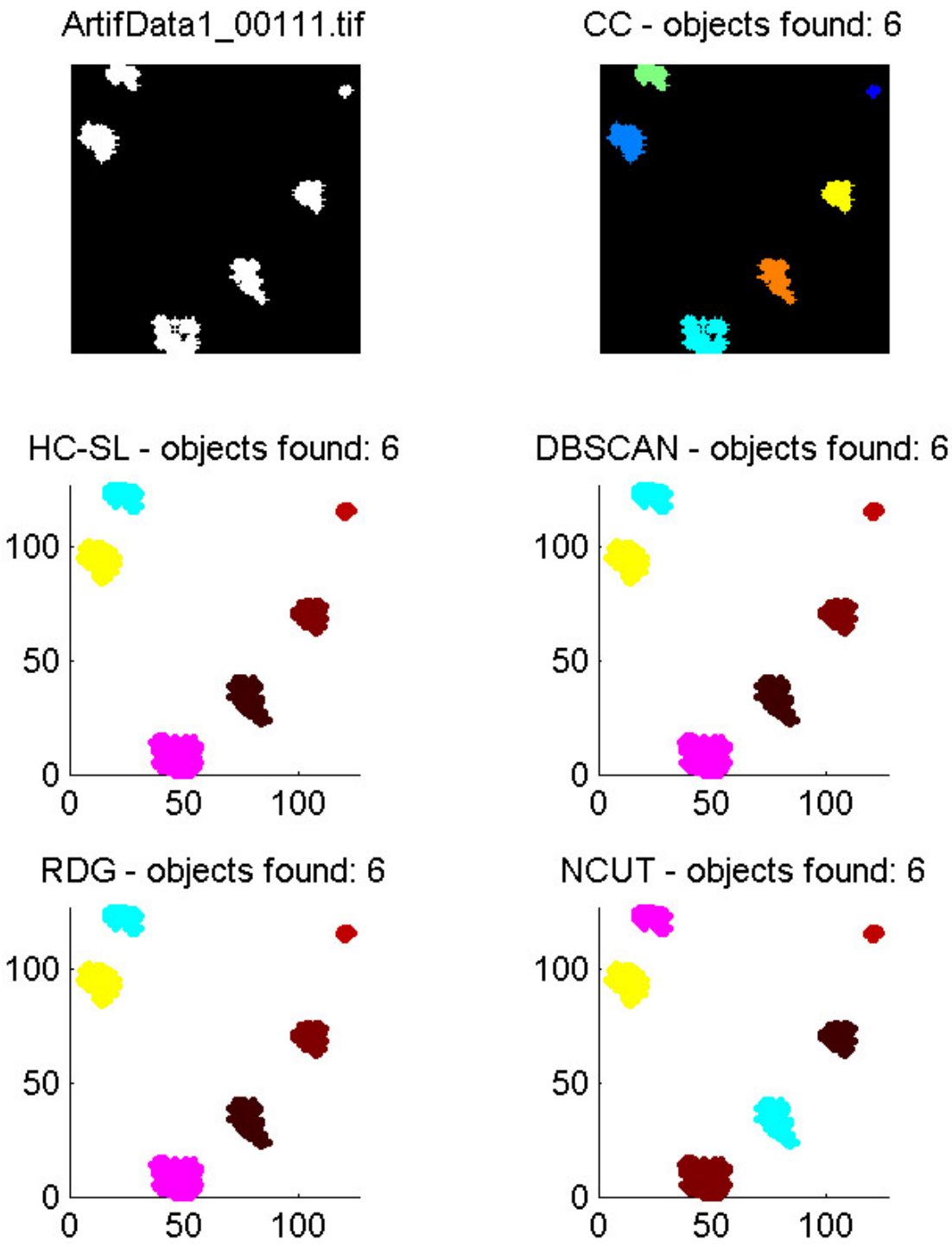


Figure 8: Here a thresholded artificial image with six objects, consisting of connected pixels, is shown. Since all clustering algorithms agree with the result of the connected component algoritm, it can be concluded that clustering is able to find compact objects as good as CC does.

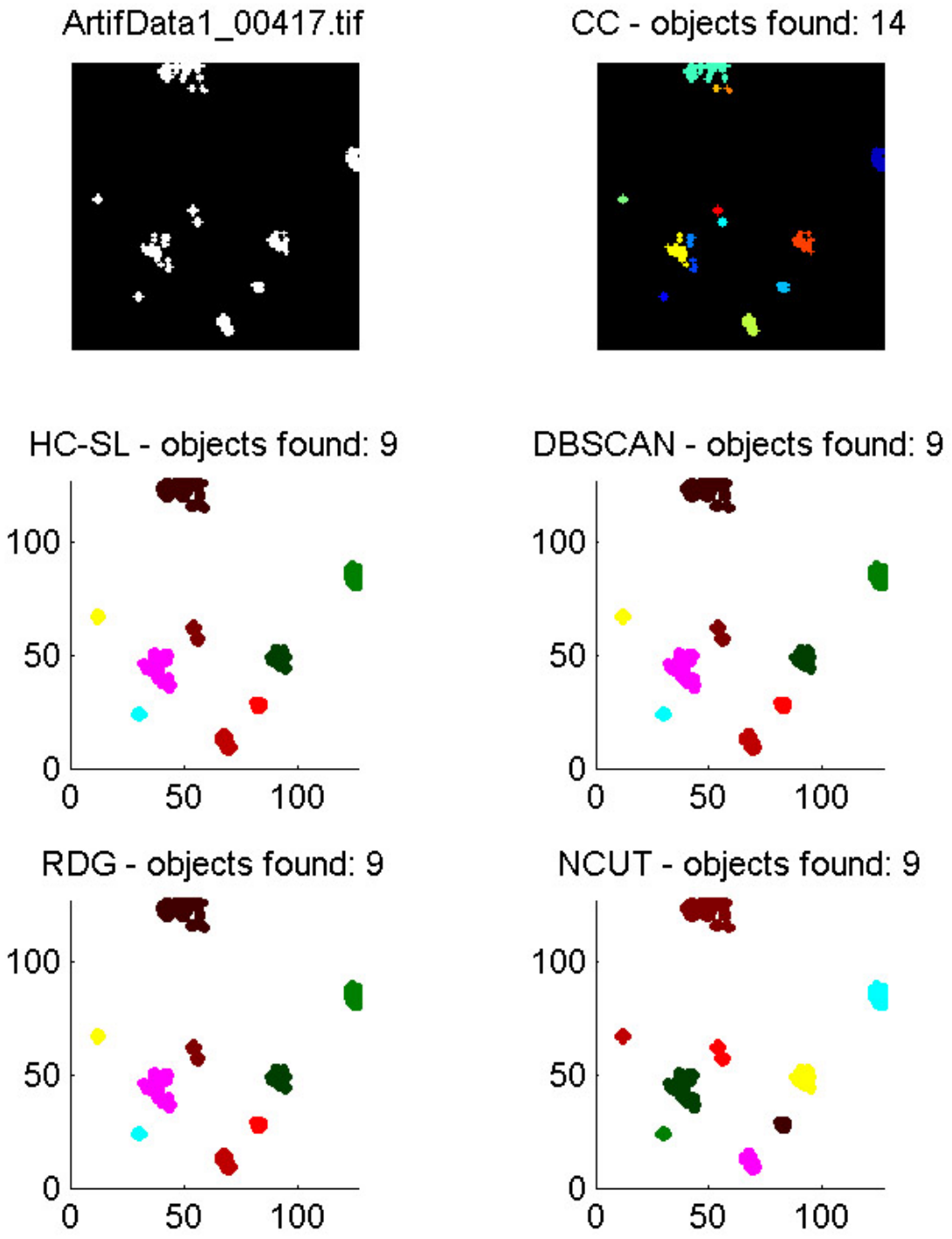


Figure 9: In the second example, the image contains nine objects. As some of them consist of pixels, which are not connected, the connected component algorithm finds too many objects. By contrast, all clustering approaches extract the correct number of objects.

3.2 Egg Inspection

Overview

In this real 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 grayscale. 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. Due to the nature of dirt and yolk, their shape can be arbitrary. The correct number of objects in the images is unknown.

Evaluation

Here the evaluation of the clustering approaches was carried out by visual inspection of the results. On the following pages, two examples are given to demonstrate the object extraction performance of the clustering algorithms. The parameters were set to fixed values or determined by automatic procedures.

Conclusion

As the object shapes are rather simple (similar to the artifical images from the previous section), all clustering methods produce meaningful results and again do better than the connected component algorithm.

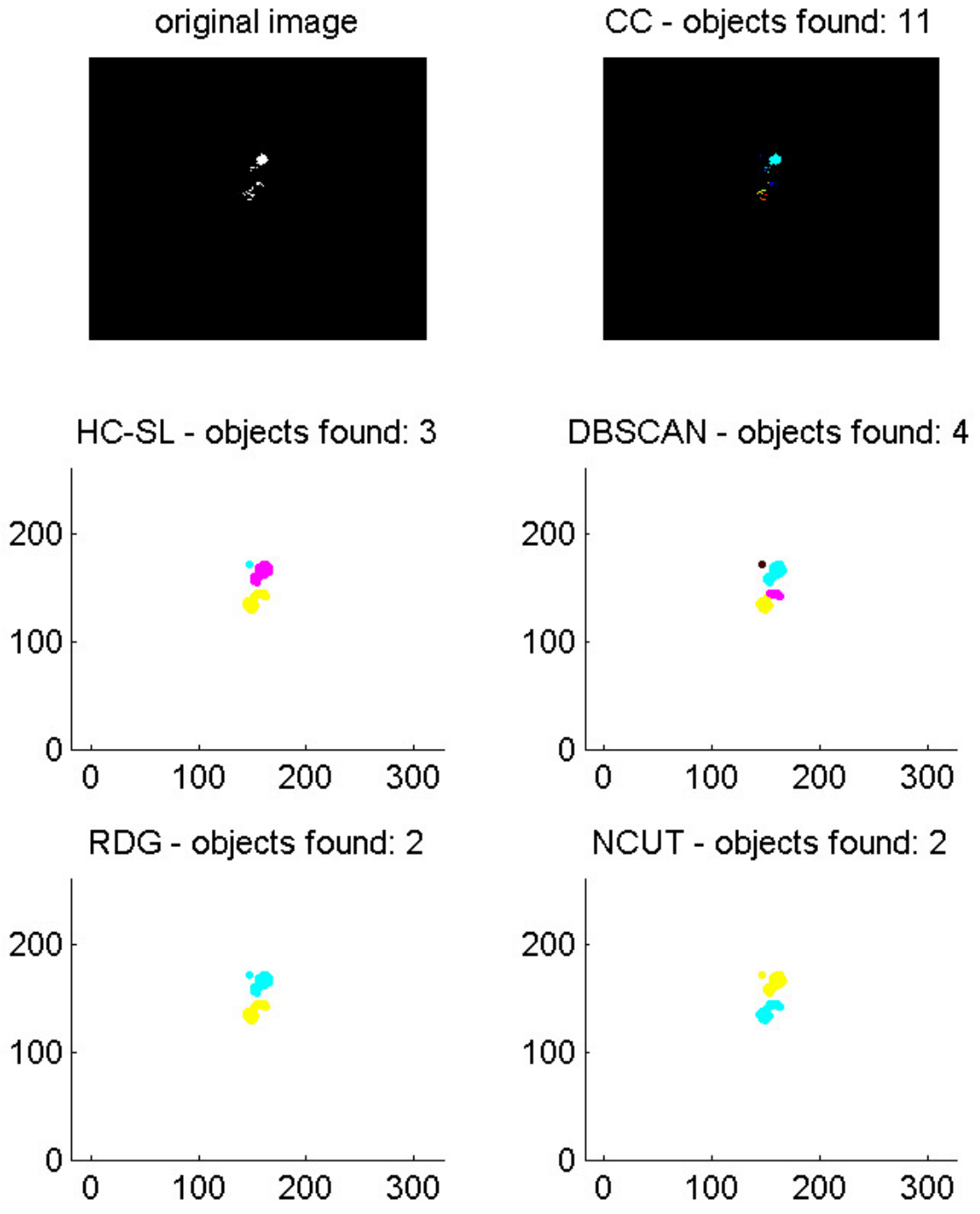


Figure 10: In this case, yolk covers a part of the egg. As the defilement is unconnected, CC finds a too large number of objects. By contrast, the clustering algorithms show better results. HC-SL and DBSCAN return a rather fine clustering, whereas RDG and NCUT produce a coarse one.

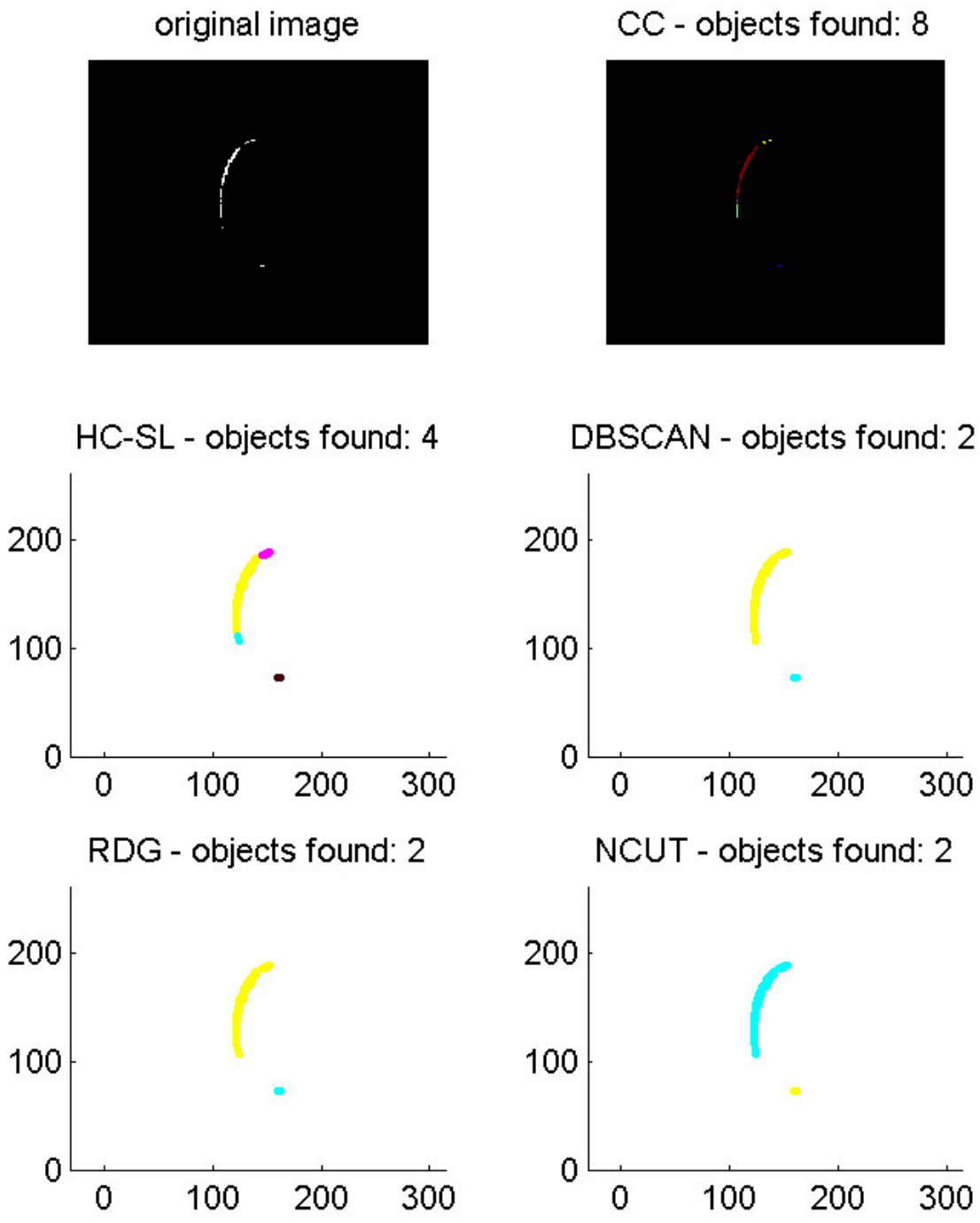


Figure 11: In this image, the challenge is to extract the arc-type structure at the egg's border correctly. As it consists of non-connected pixels, the connected component algorithm fails here as expected, but three of the clustering algorithms are able to successfully detect this thin, curved object.

3.3 CD Imprint Inspection

Overview

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 and each deviation is marked as a probable defect. The outcome is a grayscale image of size 768x576 pixels, where the graylevels correspond to the amount of deviation from the master. Obviously, the defective areas have arbitray size and shape and often exhibit some kind of scattered structures, i.e. like they occur in the case of ink splashes. After a (hopefully) successful object extraction, features are calculated for the objects found. They serve as input for a classfier, which decides whether the imprint is okay or has to be rejected. The whole process from image aquisition to the final classification result takes place in less than half a second.

Evaluation

Before clustering could be performed, the grayscale images were binarzed with a threshold operation using a threshold value of zero. Because the image size is quite high, the amount of white pixels in the binary image sometimes gets too large for the current implementation of the clustering algorithms. In these cases, a random selection of points is applied prior to the clustering.

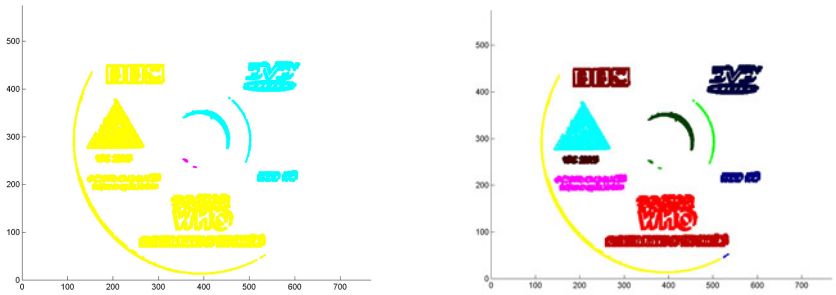
The parameters of the algorithms again were set to fixed values or determined by automatic procedures.

As the number of objects is unknown, the object extraction performance was judged by visual inspection of the results. In many cases even an expert was not able to decide which clustering performs best, since various ways of grouping pixels seemed to be meaningful.

It turned out that the choice of the parameter values strongly influences the quality of the object extraction. Figure 12 shows the impact of varying the $cutoff$ value when using hierarchical clustering. In Figure 12a the $cutoff$ was automatically set according to a rule of thumb, which leads to three objects. When the $cutoff$ is changed manually to a higher value, much better results were obtained as it can be seen in Figure 12b.

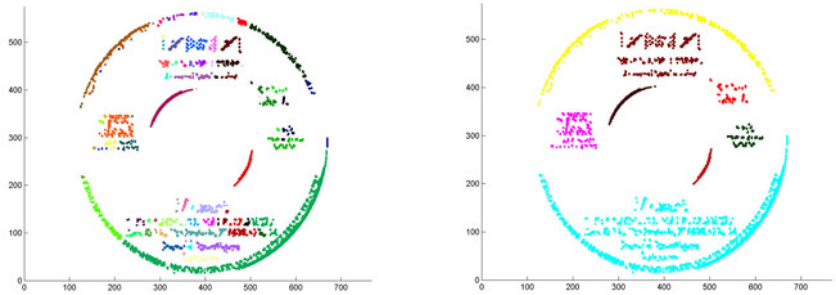
Also the automatic setting of DBSCAN's ϵ parameter via an estimation formula produces sometimes very unsatisfying results, as illustrated by Figure 13. When the value is set according to the formula, far too many objects are found (figure 13a). After a manual tuning of ϵ, the object extraction results become nearly perfect (figure 13).

The same observations were made for the parameters of the RDG algorithm.



(a) (b)

Figure 12: Using HC-SL with a) automatic $cutoff = 44.8$ (3 objects) and b) manual $cutoff = 8.0$ (14 objects).



(a) (b)

Figure 13: Using DBSCAN with a) automatic $\epsilon = 7.6336$ (97 objects) and b) manual $\epsilon = 20.0$ (8 objects).

On the following pages two examples are given to demonstrate the object extraction performance of the clustering algorithms on imprint images with defects of various shape.

Conclusion

For the CD imprint images, the connected component algorithm fails completely, because the objects in this application are highly discontinous and scattered. By contrast, all clustering algorithms presented here, have the potential to perform a meaningful object extraction even in this tough case. However, in practice the applicability of these approaches depends strongly on how an appropriate parameter setting can automatically be determined for each image.

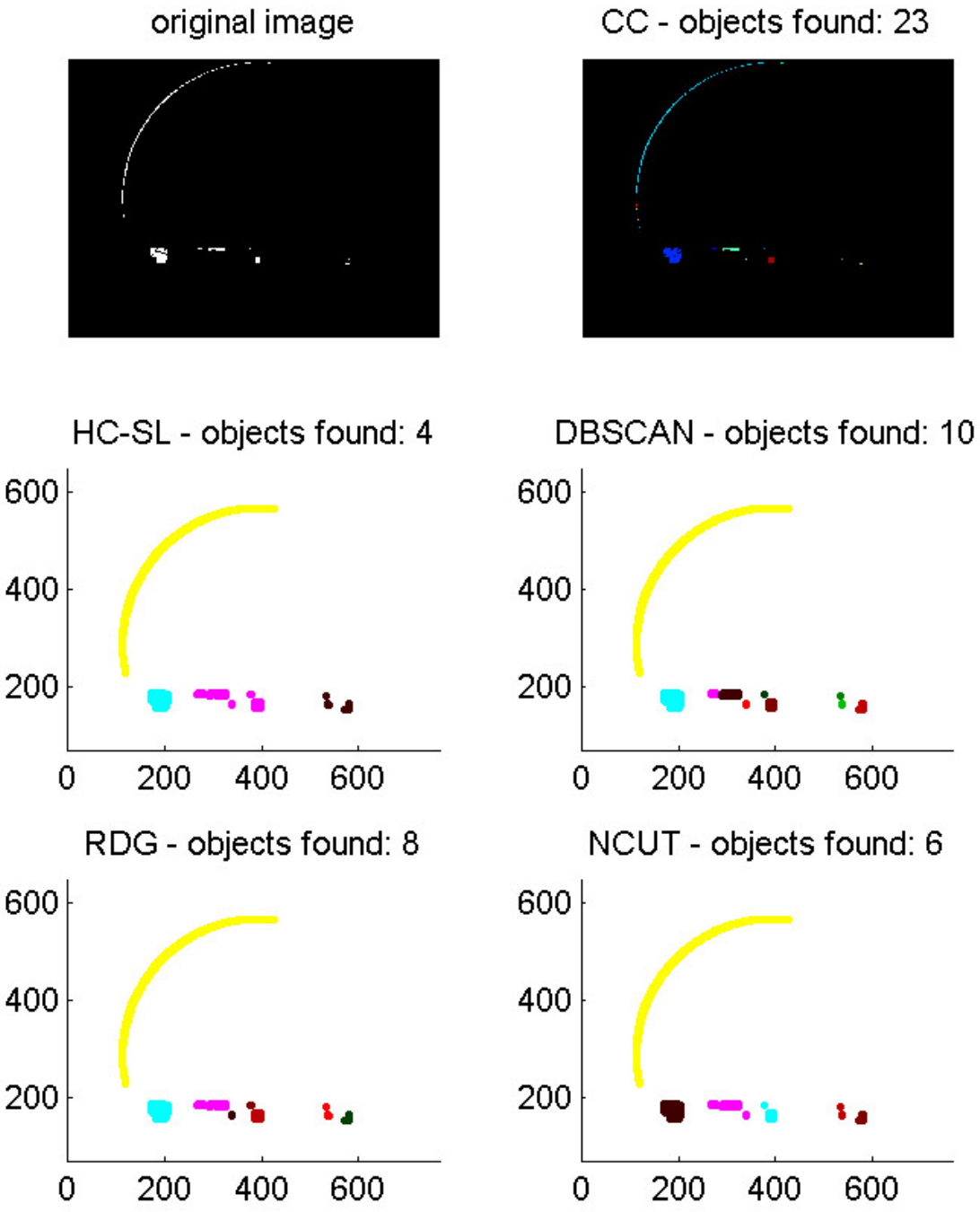


Figure 14: In this case all clustering algorithms work well, but produce different pixel groupings from very fine (DBSCAN) to coarse (HC-SL). Which result is best, is a matter of argumentation. The arc at the border of the CD is extracted perfectly by all methods. Again, the connected component algorithm returns worse results.

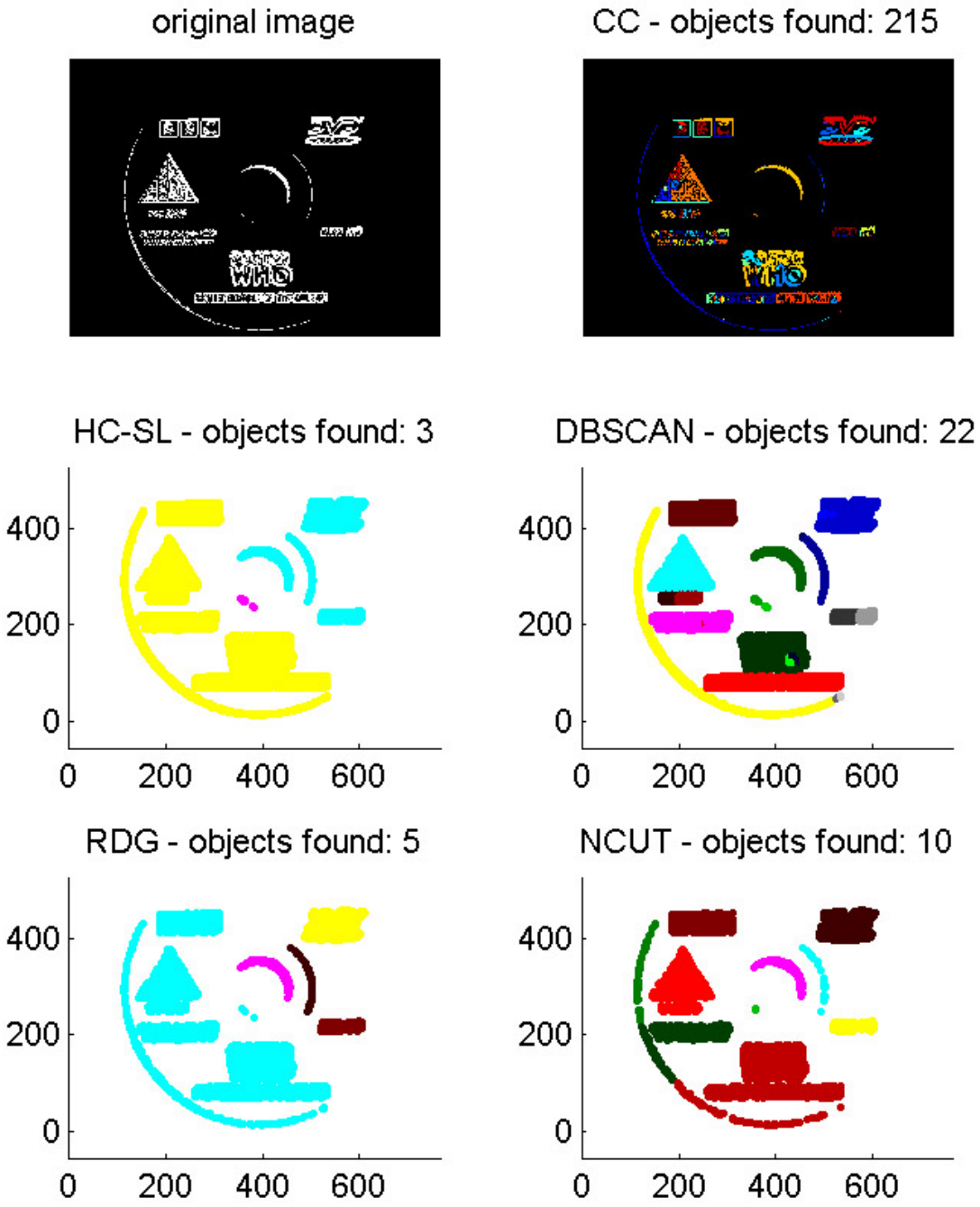


Figure 15: This image contains many objects of different shape. Only DBSCAN and NCUT produce satifactory results, while HC-SL and RDG seem to fail here. As previously mentioned, this is caused by an improper automatic parameter setting. If the parameter values are tuned manually, both approaches can keep up with DBSCAN and NCUT (see Figure 12). As expected, the connected component algorithm identifies too many objects.

3.4 Thin-Film Sensor Inspection

Overview

The last application deals with images from the prototype of an automatic inspection system for thin-film sensors, where defects like holes or scratches have to be identified. Here, a grayscale line camera with 2048 pixels, mounted on top of a microscope, together with a motor-driven scanning stage represent the main components of the image aquisition equipment. As light source, the bright field illumination of the microscope is used. Again, the obtained images are compared with an image of a fault-free master part, in order to detect any deviations. Like in the case of the CD imprints, the objects to be found can have arbitrary size and shape. When a defect, i.e. a scratch, streches out across different sensor structures, it is very often split up in a couple of pieces, which have to be grouped together again by the object extraction algorithm.

Evaluation

As the original images from the inspection process are very big (up to 1500x9550 pixels[1]), the resolution was decreased (down to one-third) and only a small (interesting) area of 256x256 pixels was selected for evaluation. The thresholding operation to binarize the grayscale images, was carried out manually, in order to suppress any undesirable noisy artifacts.

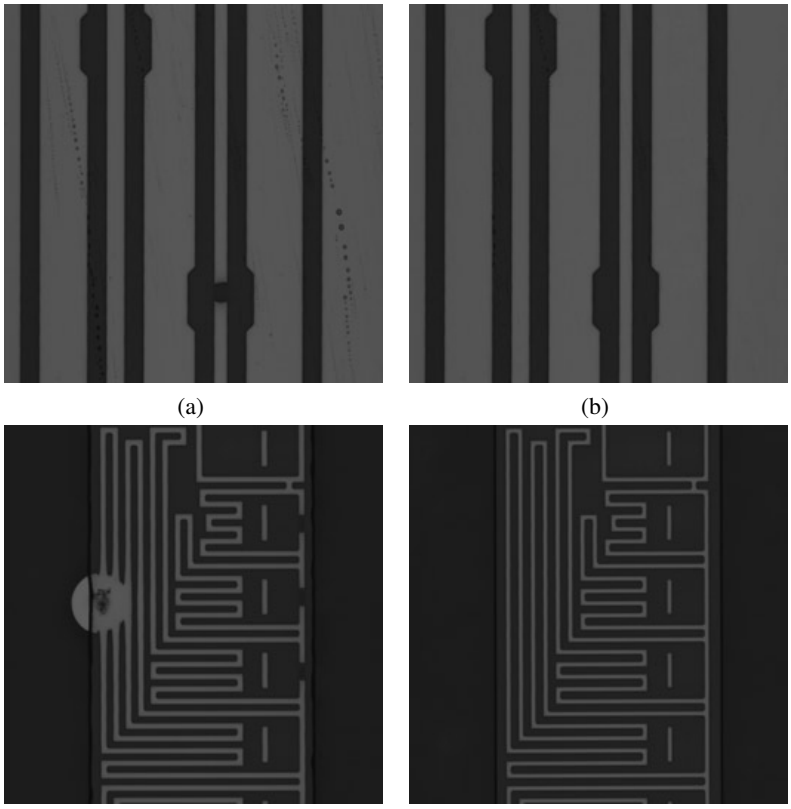
The parameters of the clustering algorithms were set to fixed values or determined by automatic procedures.

Figure 16 gives an overview on the two example images, discussed on the following pages. The left column shows the (probably) defective test images, the right column illustrates the corresponding fault-free master images. As the number of objects is unknown, the object extraction performance was again judged by visual inspection of the results.

Conclusion

Due to the huge variety of possible defects, only some representative cases could be investigated. But the results obtained so far, seem very promising. The clustering algorithms again outperform the simple connected component approach and are able to produce resonable groupings, if the parameters can be determined correctly during on-line operation.

[1] Not the full width of the line camera (2048px) is used here because of vignetting.



(a) (b)

(c) (d)

Figure 16: The two test images (left column) with their corresponding master images (right column).

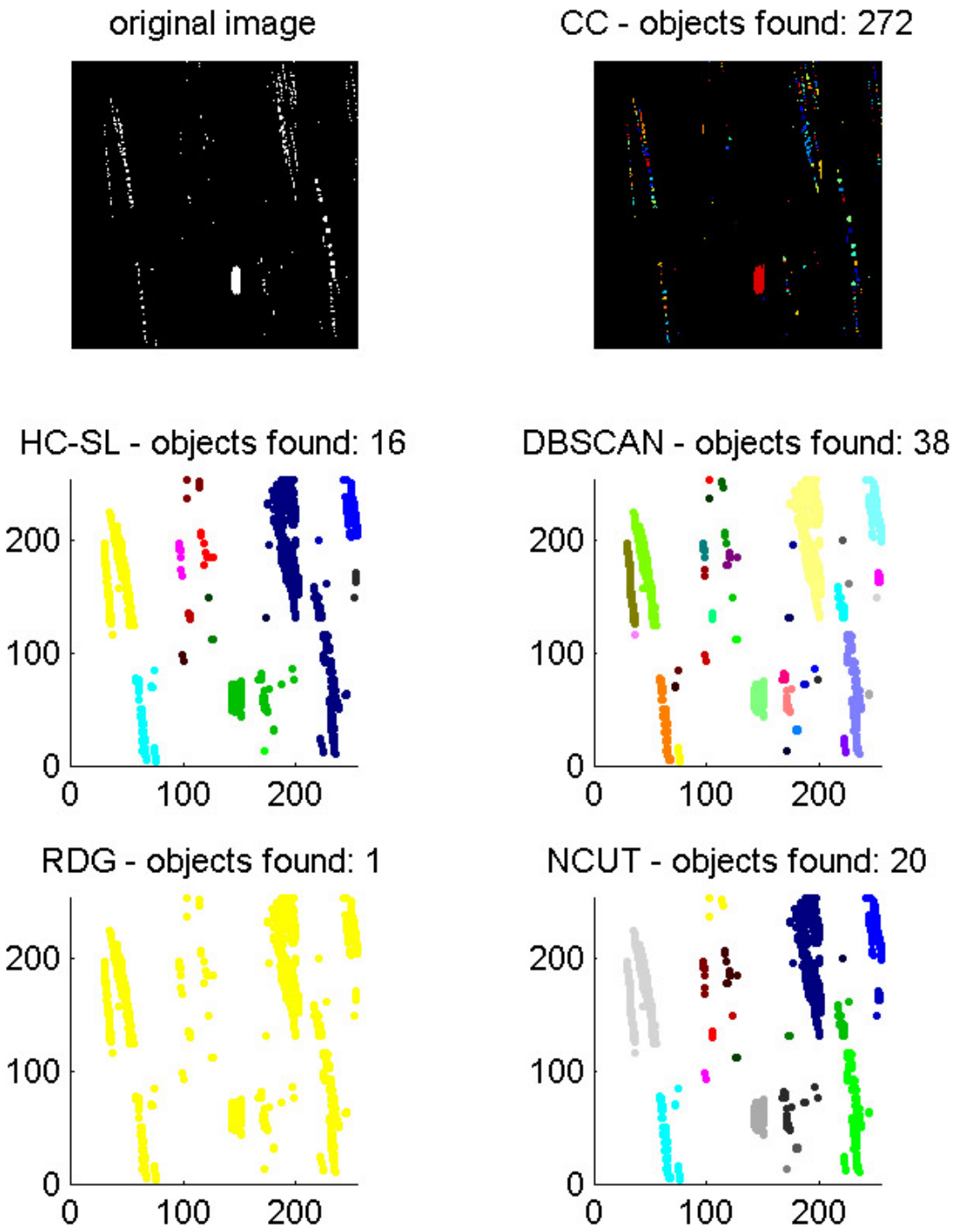


Figure 17: Here, mainly tilted, discontinuous lines are present together with a single compact object in the lower middle. Except RDG, which obviously fails to automatically find the right threshold, all clustering approaches produce meaningful groupings, while the connected component algorithm returns far too many objects.

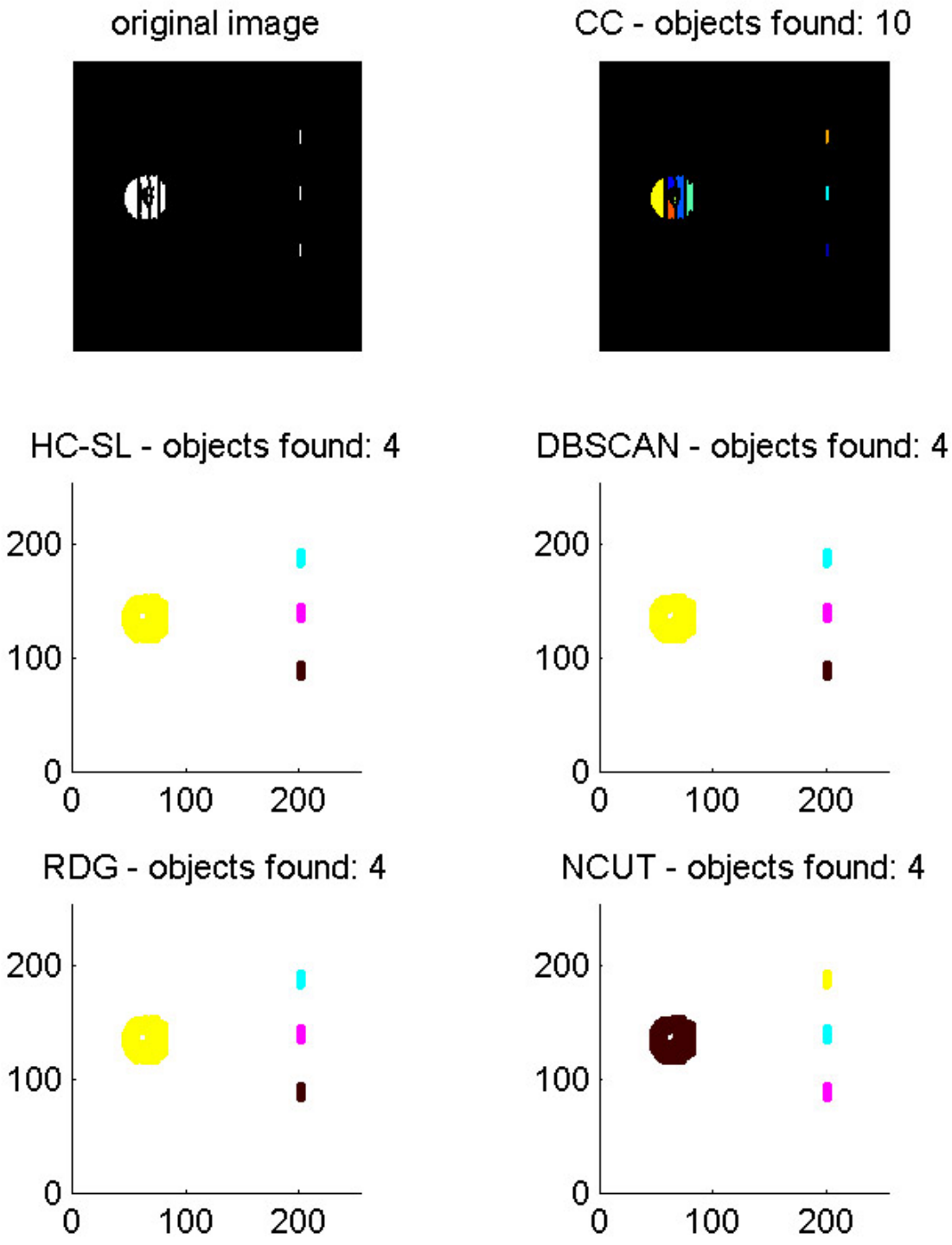


Figure 18: The last image contains three small compact objects on the right and a single blob-shaped object on the left, which has been split up into serveral pieces. Of course, the connected component algorithm is not able to merge the individual pieces together. But all clustering approaches recombine correctly the parts of the blob into a single object.

4 Conclusion and Outlook

In this paper the applicability of clustering methods for object extraction from images, as they occur in industrial machine vision applications, has been investigated. After an overview about the object extraction task and its specific challenges, a possible way to incorporate clustering methods in the object extraction process has been presented. Starting with cluster tendency analysis to verify the existance of objects in an image, followed by clustering algorithms for the object extraction itself, the discussion finally leads to cluster validation techniques used for determining optimal parameter settings. After that, most of the methods presented, have been evaluated on artificial as well as on real images.

When using clustering methods in this field of application, one has to deal with two major challenges: their high computational cost (depending on algorithm and implementation) and the problem of automatically finding an optimal parameter setting. Especially, the latter turns out to be of utmost importance, since the performance of most of the clustering algorithms depends strongly on the choice of their parameters. So, future work will have to put its main focus on these two issues.

Also the object extraction quality still can be improved by enhancing the clustering itself. In the following, a selection of ideas is listed:

- Extension of the data set to be clustered (currently consisting only of pixel coordinates) by adding extra information like the pixel's color value in the test and/or master image.
- Switching to object features instead of pixel coordinates. Here, after inital objects are identified by CC, features like the center of gravity, orientation, roundness, etc. are calculated and used for clustering.
- Incorporation of application-specific background knowledge in terms of clustering constraints [13].
- Combination of different clustering results using so-called cluster ensembles [14].

To summarize, using clustering methods for object extraction, especially when the objects to be found are disconnected and wide-spread, seems to be a very promising and feasible approach.

References

[1] L. G. Shapiro and G. C. Stockman, *Computer Vision*. New Jersey, USA: Prentice Hall, Inc., 2001.

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

[3] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[4] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[5] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data*, 2006, pp. 25–71.

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

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

[8] R. Sibson, "Slink: An optimally efficient algorithm for the single-link cluster method," *Computer Journal*, vol. 16, no. 1, pp. 30–34, 1973.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 226–231.

[10] 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*, vol. 3704. Springer-Verlag Berlin Heidelberg, October 2005, pp. 497–506.

[11] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[12] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *Journal of Intelligent Information Systems*, vol. 17, no. 2/3, pp. 107–145, 2001.

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

[14] A. Strehl and J. Ghosh, "Cluster ensembles — a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, December 2002.

Clustering Terahertz-Spectra and Text-Documents

Finding a Common Approach to Different Kinds of High-Dimensional Data Sets

Henrike Stephani
Fraunhofer Institute for Industrial Mathematics (ITWM)
Fraunhofer Platz 1
67663 Kaiserslautern
Henrike.Stephani@itwm.fraunhofer.de

Abstract

There is a broad variety of clustering algorithms applied on both text as well as spectroscopic test data. Hierarchical clustering methods hold the advantage of good interpretability and choice of depth of clustering. The two areas, text mining and spectral analysis are rather distinct but both high dimensional in the feature space. It is tried in this short paper to enlighten the possibilities of common methods.

1 Introduction

With increasing data volume available, the task of organising it in a sensible way is becoming more and more important. Often the supplied data is unlabelled as it is easier to obtain data than to label it. The latter generally requires a certain amount of expertise which is not always at hand. Therefore, coping with unlabelled information shall be the issue here.
In machine learning organising unlabelled data is called unsupervised classification or clustering as opposed to supervised classification. In supervised classification some training data is required, i.e. the classes and a sufficiently big amount of samples from these classes have to be known beforehand to build the classifier. The most prominent among the applied algorithms in this area are decision trees and Support Vector Machines (SVM). [4].
On the other hand in unsupervised classification the classes are not known, neither in shape, nor position, nor number [11, 2, 20]. All this information needs to be derived from the clustering algorithm. There is a great variety of algorithms for unsupervised classification such as K-means, neural networks, single and complete link, and more. Considering the multitude of different kinds of data, unsupervisedly finding these inherent characteristics and groupings is no easy task. Therefore like in supervised learning expertise is needed. It is put into deciding which method to use for which set of data or how to preprocess the data in an appropriate way [20, 11].
Another approach to classification is to not sharply divide the algorithm in supervised and unsupervised learning but to leave room for semi supervised methods [9]. In this approach, a certain amount of labelled data is given but relatively small in comparison to the unlabelled patterns. Research is being done on how to adapt existing algorithms to this problem. As a result there are for example transductive SVM [12]. It is a drawback though, that usually it is presumed that the labelled data contains at least one sample from each evolving class.

In this paper ideas on how to cluster different sets of data are developed. On the one hand there is clustering of spectral data obtained from Terahertz measurements of chemical compounds. On the other hand high volume sets of text documents. In both cases have the high dimensionality in common, therefore in introducing some basic clustering methods in this paper, there shall be a focus on methods that are applicable on high dimensional data.

2 Methods

To build a sensible and comparable method for data grouping, the clustering algorithm itself is only one step in a process. In general the steps are divided into feature selection, finding an appropriate proximity measure, the clustering, and evaluation of the clustering [20, 11]. The steps shall be briefly described.

2.1 Feature Selection

Feature selection consists of deciding which are the relevant features to represent the given pattern. For example it has to be considered whether to use binary or metric at-

Figure 1. Steps of K-means Algorithm.

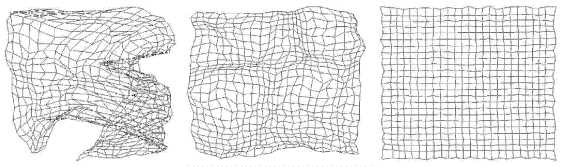


Figure 2. Random Net of Nodes Self Organised by Uniformly Distributed Input Patterns.

tributes. Taking into account how the patterns are obtained further preprocessing or denoising might be sensible [19]. If the dimensionality is very high, means of reducing the dimensionality should be taken [14]. Preprocessing takes an important role within the clustering task and usually requires some kind of expert knowledge. However it is very dependent on the application [15].

2.2 Proximity Measure

As in clustering one seeks to group similar things close together and different things far apart, so a measure of similarity or distance has to be defined. The search of the right similarity measure varies in importance depending on the clustering algorithm and therefore is often combined with the search of the clustering algorithm [20].

2.3 Clustering Algorithms

This work shall focus on this aspect. A short review of the relevant methods shall be given here.
There is a big variety of clustering methods in use and an even bigger variety being developed. In literature methods are usually categorised at least in hierarchical and partitional clustering.

2.3.1 Partitional Clustering

In partitional clustering the goal is to distribute the patterns on different classes. Based on some distance measure the data is assigned to its nearest cluster. The most important representative of these algorithms is the K-means algorithm. In its classical version, illustrated in figure 1, the number K of clusters has to be known beforehand. The initial K cluster centres are distributed randomly over the feature space and the patterns assigned to their closest centre. Of the obtained clusters new centres are calculated and the whole process of assigning the patterns to the new cluster centres iterated and so on.

There are different implementations of this algorithm. There are also versions in fuzzy clustering [5]. Although in worst case scenarios K-means has exponential performance [1] in average case it works quite well (complexity $O(NK)$) and gives good results [3] also in high volume data sets. Furthermore it is quite easy to implement but there are some drawbacks. The strong dependence on the similarity measure leads to a confrontation with the "curse of dimensionality" [2]. The algorithm is sensitive to outliers and unstable. Also the choice of the initial cluster centres is an important issue of research. Most of these drawbacks have been worked on. In general, K-means is one of the best researched clustering algorithm.

Artificial neural networks are an attempt to find an anologon to neurons in the human brain which are capable of coping with high dimensional data mapped on a one-dimensional neural node structure. There are different ways to initialise such a neural network. One important way are the Self Organizing Maps (SOM) [14, 3]. Here a one or two dimensional net of a certain size is initialised with weights attached to each node. The weights have the same dimension as the features of the input data. One by one the input data is mapped onto the nearest node. The node and its neighbours are then assigned new weights depending on the feature values of the assigned input pattern. This is done in many iterations and results in a net structure with different concentrations. In figure 2 it is shown how the aspect of a randomly given net iteratively changes in SOM. Another important algorithm is the perceptron algorithm [16]. It usually consists in an input layer of nodes, a hidden layer, and an output layer. The feedback, i.e. the reweighing, is done in the hidden layer.

2.3.2 Hierarchical Clustering

In hierarchical clustering the given patterns are initially either all assigned to one group or each one to an own group. They are then getting clustered by splitting (divisive) the farest apart or uniting (agglomerative) the closest classes until finally all patterns are in seperate classes or all in the same class respectively. The basic agglomerative methods are single link and complete link. In these methods the distance between two clusters is defined as the minimum or the maximum distance of its cluster points respectively [2, 11]. As a result of hierarchical clustering one gets a so called dendrogram, as is shown in figure 3, which shows the point of the algorithm where patterns are united or divided.
These algorithms work well for different kinds of features and similarity measures, but it is hard to find a termination criterion and usually the results are not revisited [6]. Also

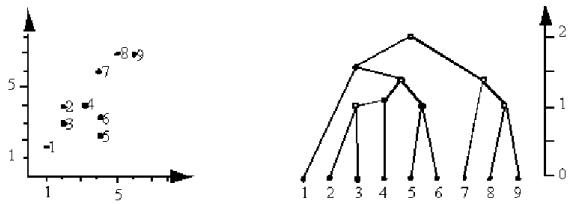


Figure 3. Patterns Clustered by Single-Link Method to a Dendrogram.

in classical methods the scalability is quite bad.
In grid based methods the main idea is to put a grid over the input data and eliminate grid cells which contain less data points than a certain threshold. By merging neighbouring cells again, the clusters are obtained. The basic grid based method suffer from the curse of dimensionality as the data often is too sparsely distributed in some areas. The hierarchy is achieved by using different sizes of grid cells.
A relatively new approach in hierarchical clustering is the Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) Algorithm [21]. It is especially designed to handle large data sets. It puts a major part of its calculatory cost in the data uploading phase. In this phase the data is grouped in a feature tree structure in dense sub clusters which represent the data in the further process. The following clustering algorithm usually is agglomerative hierarchical [20]. Thus it is fast, can handle noise and does not require much memory.

2.4 Evaluation

In nearly all these methods, some more than others, several problems have to be faced. In clustering in general there is the problem of scalability, order dependence, the validation of the clusters, the curse of dimensionality, the computational complexity of the algorithms and how they deal with new data i.e. if incremental work is possible [18].
Thus a big part of evaluating the clustering is judging if the algorithm works, i.e. if it gives any formally sensible result and how fast this result is given. If the result, when achieved, is correct, is usually evaluated by testing the method with labeled data.

3 Data

During this work, two sources which will provide data samples are considered. On the one hand there is a cooperation with a governmental program, called ORDO (Organize your Digital Life). Other data will be provided by the Fraunhofer Institute for Physical Measurement Techniques (IPM).

3.1 ORDO

In this project it needs to be dealt with text documents like for example patent data. Organising big amounts of text documents is known as data mining. Mostly statistical methods are used to achieve a structure. Here this task shall be looked upon as a case of high dimensional data clustering. The dimensionality is given by the amount of used words. In working with the high volume raw data, it has to be coped with 10.000s of dimensions. There have been different attempts to handle this amount of data.
K-means is a common device. It also has been tested on unpreprocessed patterns. Without any dimensionality reduction key words of a set of 6.000 words could be identified [11]. In another approach an adaptation of the SOM has been developed to cope with high dimensional high volume data [14]. This was tested on a data set of more than 6.000.000 patent abstracts. Preprocessing like dimensionality reduction had to be done. The resulting map had an accuracy of about 60% which requires human post processing.
Nevertheless this shows that the SOM algorithm works on such data amounts and gives evaluable results. Grid based methods, like BIRCH, are not so easily applicable, because they usually require a metric input space [21, 2], which is not given here.
Hierarchical methods, as mentioned before, have the disadvantage of bad scalability. A new promising method in clustering seems to be the Frequent Itemset based Hierarchical Clustering (FIHC) [8] where frequently appearing itemsets are identified and the patterns are assigned to the itemset which they contain with the highest score. Finding the itemsets is a well researched method from data mining. The further clustering works with hierarchical method. FIHC was evaluated on common test data and emerged as being more accurate, efficient and scalable than its competitors in clustering high dimensional, high volume text documents [7].

3.2 IPM

The Fraunhofer IPM investigates in the relatively new area of Terahertz spectroscopy. Terahertz waves cover the frequencies between infrared and microwave, that is 0.3 to 20 THz. For a long time it was not possible to measure these waves due to lack of power or sensitivity of transmitter and receiver respectively. Terahertz waves will probably increase in importance over the next decades. Reasons for their importance are, that they are non ionising which means that they do not change or influence the materials which are exposed to them. The radiation is not absorbed by cloth, paper, wood, plastic or ceramics. The last aspect

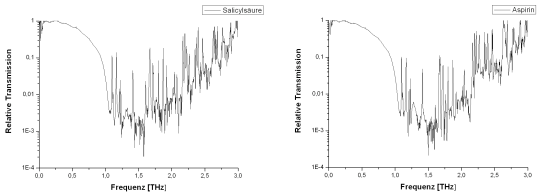


Figure 4. Spectra of salicylic acid and Aspirin. Fourier transformed and plotted relative to a reference measurement. Both spectra seem quite similar. In quality control one would want a machine to be able to differentiate these two, i.e. to find characteristics which Aspirin do have and salicylic acid does not.

makes it especially interesting for security and quality control. On the other hand absorbing materials are metal and water. Another aspect is that chemical compounds do have a characteristic absorbance spectrum. Therefore it is theoretically possible to detect chemical compounds in an item without destroying or changing it. For further information on the chemical and physical sides of Terahertz measurements see [10, 17].

The cooperation with the IPM will focus on latter aspect. The relevant spectra have not yet been all measured and even less been categorised due to the newness of the methods. Until now, they mainly get processed by mere identification and manual comparison of the major peaks. The goal of a cluster analysis would be to identify categories of compounds and finally the compounds themselves. It is important to use methods that are working incrementally, as there will be new data obtained constantly.
Several methods have been applied to cluster spectral data. Before the actual clustering can take place there usually is some preprocessing done to reduce noise and get better results. Instead of using the raw data, some Fourier or wavelet based noise reduced data is considered [19]. Noise reduction will be of special importance as Terahertz radiation is absorbed by water and thus very sensitive to humidity in the measure space. Clustering these spectra is a special case of clustering time series data. In general, time series data does not differ much from general high dimensional data. The special characteristics of it being time series data are put into the preprocessing and feature selection [15]. So similar methods as described above are applied. A difference to text document clustering is that spectral data features are not binary but metric. Nevertheless the feature selection is one of the essential parts here as well. Usually the considered features are the peaks of the spectra. To find relevant ones there are different algorithms in use. For further information on that see [13].

4 Further Work

There seems to be a general agreement in research on unsupervised classification, that every test case needs to be treated with a special algorithm. Although many algorithms are used for different kinds of data, there is still a great lack of comparison between these methods. Therefore the area of methodical investigation of differences, advantages and drawbacks of the methods needs to be further researched on.
It should be surveyed whether there exists a possibility to shift the special needs of the different data sets to preprocessing and feature selection. Therefore it will be tried to regard both the IPM and the ORDO test data as special cases of high dimensional samples.
We will focus on hierarchical clustering methods, as the coarseness of the clustering can be influenced and the amount of classes does not have to be determined before. Another aspect is, that there are promising developments in coping with scalability as for example the BIRCH and the Frequent Itemset based Hierarchical Clustering (FIHC) algorithms. In the latter it could be tried to find relevant peaks as much as a analogon to the frequent items used in FIHC. The research on finding peaks is a topic in mass spectroscopy already [13].
The missing possibiliy to work incrementally is still the major drawback of hierarchical clustering [8]. The possibilities in this area should also be further investigated.

References

[1] D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153, New York, NY, USA, 2006. ACM.

[2] P. Berkhin. *Grouping Multidimensional Data*, chapter A Survey of Clustering Data Mining Techniques, pages 25–71. Springer Berlin Heidelberg, 2006.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[5] C. Doring, M.-J. Lesot, and R. Kruse. Data analysis with fuzzy clustering methods. *Computational Statistics & Data Analysis*, 51(1):192–214, November 2006.

[6] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[7] B. Fung, K. Wang, and M. Ester. Hierarchical document clustering using frequent itemsets. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 59–70, 2003.

[8] B. C. M. Fung, K. Wang, and M. Ester. *The Encyclopedia of Data Warehousing and Mining*, chapter Hierarchical Document Clustering, pages 555–559. Idea Group, Hershey, PA, July 2005.

[9] L. Grlitz. *Modern Concepts for Semi-Supervised Learning and Multidimensional Image Processing*. PhD thesis, Ruprecht-Karls-Universitt Heidelberg, 2007.

[10] M. Herrmann, R. Fukasawa, and O. Morikawa. Terahertz imaging. In K. Sakai, editor, *Terahertz Optoelectronics, Topics Appl. Phys.*, volume 97, pages 331 – 382. Springer-Verlag Berlin Heidelberg, 2005.

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

[12] N. Kasabov and S. Pang. Letter transductive support vector machines and applications in bioinformatics for promoter recognition.

[13] M. Kirchner. *Analysis of Spectral Data*. PhD thesis, Ruprecht-Karls-Universitt Heidelberg, 2008.

[14] T. Kohonen. Self-organization of very large document collections: State of the art. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of ICANN98, the 8th International Conference on Artificial Neural Networks*, volume 1, pages 65–74. Springer, London, 1998.

[15] T. W. Liao. Clustering of time series data - a survey. *Pattern Recognition*, 38:1857 – 1874, 2005.

[16] R. Rojas. *Neural networks: a systematic introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996.

[17] C. Schmuttenmaer. Exploring dynamics in the far-infrared with terahertz spectroscopy. *Chemical Reviews*, 104(4):1759–1780, 2004.

[18] M. Steinbach, L. Ertös, and V. Kumar. The challenges of clustering high dimensional data. In L. T. Wille, editor, *New Directions in Statistical Physics : Econophysics, Bioinformatics, and Pattern Recognition*, pages 273–307. Springer, Berlin, 2004.

[19] Y.-P. Wang, Y. Wang, and P. Spencer. Fuzzy clustering of raman spectral imaging data with a wavelet-based noise-reduction approach. *Appl. Spectrosc.*, 60(7):826–832, 2006.

[20] R. Xu and I. Wunsch, D. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645–678, May 2005.

[21] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114, 1996.

Opacity Quantification In Cardiac Angiogram Sequences

Kazmar T

June 16, 2008

Abstract

The level of perfusion is routinely analyzed from the level of opacity in cardiac X-ray angiograms. We propose an image enhancement method for angiogram sequences by motion compensation and background subtraction. Moreover, we extract a time-series describing the opacification in a given region-of-interest and quantify perfusion by a toxicological model to automatically assign TMP opacity level values [4]. The effect of background subtraction to angiogram enhancement is tested as well as the performance of region tracking based on non-rigid alignment, or the model used for quantification. The tests are performed on clinical data.

1 Introduction

We analyze a cardiac X-ray angiogram sequence in order to quantify microvascular injuries resulting from myocardial infarction on the TMP scale. The aim is to develop a simple to use semi-automatic or fully automatic method to accomplish objective measurement of perfusion, as opposed to currently used observer-dependent visual inspection.

Cardiac angiogram sequences have four different phases [1] which are: *inflow* when the dye enters the arteries, *complete state* when the arteries are fully opacified, *washout* when the dye leaves the arteries and *venuous phase* when it enters the veins. An example of an input image can be seen in Figure 3 (left). For our algorithm we need at least the first three phases to be present in the data. During the inflow or the complete state phases the opacity of microvasculature increases, and depending on how much and how long the opacity persists we can determine the level of perfusion for small vessels and thus also the level of damage. In this work we present a method to extract curves describing the time course of the opacification and apply a perfusion model to these curves.

2 Methods

The proposed method is based on image registration to compensate heart motion. Registration is followed by background estimation, search for matching cardiac cycle frames, region-of-interest (ROI) tracking and the quantification itself.

2.1 Motion compensation and background subtraction

For movement compensation, we use a non-rigid image registration method where deformation is represented as a B-spline transform [7, 8]. Multi-resolution is used in both image and transformation space to improve speed and robustness, specifically Gaussian pyramids and B-spline pyramid. The alignment error is measured using a SSD-criterium which is optimized by L-BFGS optimizer. Non-rigid registration is initialized by translational alignment which compensates the table shifts, if present.

After registration, we estimate the background as the mean intensity image over all frames in the aligned sequence:

$$\sum_{k=0}^{n} B(\vec{x}, j) = I\left(T_{j,k}(\vec{x}), k\right)$$

where $I(\vec{x}, k)$ is the k-th frame and $T_{j,k}$ is the deformation between frames j and k, found in the preceding step. Subtracting the background removes static objects and emphasizes opacity changes, see Figure 3 (right), similar to digital subtraction angiography (DSA).

Unfortunately, the registration yields alignments with varying quality. We therefore propose to take only frames which are well aligned with the current frame to get better results. A chance for a good alignment is bigger for frames pertaining to the same phase of the heart beat (systole, diastole) as the current frame – we call such frames *matching frames*. We use the fact that the heart tissue is periodically stretched and compressed which shows as different intensity levels. For each frame, we search for a set of matching frames, one in each heart beat. Heart rate is estimated from the periodicity of the image mean intensity $\sum_{\vec{x}} I(\vec{x}, j)$ denoted by r. More specifically, we estimate the heart beat frequency $\hat{f}$ as the most prominent frequency of r after applying a high-pass filter H using Fourier transformation (Figure 1):

$$\hat{f} = \operatorname*{argmax}_f R(f)H(f) \text{ where } R(f) = FFT_t(r)$$

Alternatively, we could obtain $\hat{f}$ from simultaneous ECG. However, in our case ECG data is not available.

2.2 Opacity curves and perfusion model

The opacity changes are evaluated inside the ROI $\Omega(j)$ which is set by user in one frame. Thanks to known inter-frame deformations T we can track the ROIs throughout the sequence.

$$\Omega(j) = T_{k,j}(\Omega(k))$$

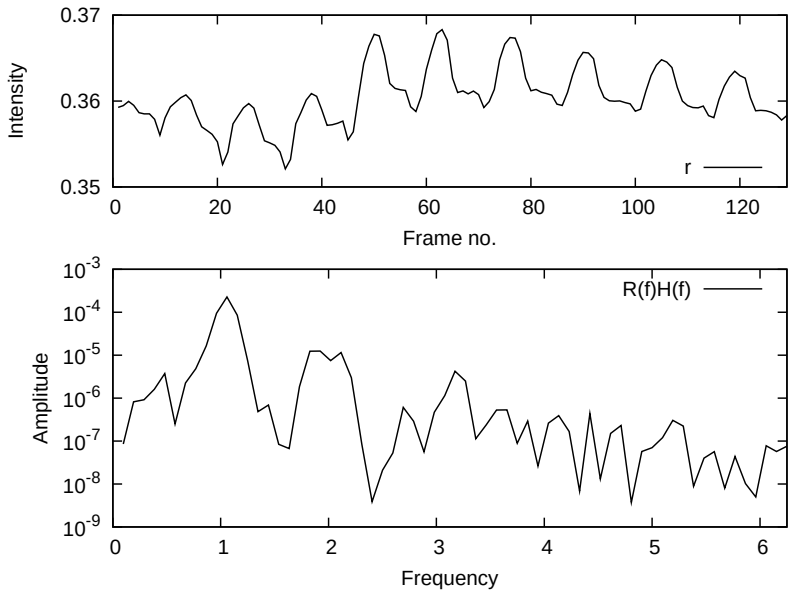


Figure 1: Heart rate estimation from the image mean intensity.

Warped ROIs can be further adjusted by the user. We have observed that it is better to use a ROI from a matching frame than from an arbitrary frame.

An *opacity curve* describing time evolution of the mean opacity in the ROI is extracted as follows:

$$q_2(j) = \int_{\Omega(j)} I(\vec{x}, j) d\vec{x} \Big/ \int_{\Omega(j)} d\vec{x}$$

By analogy we extract an opacity curve of arteries q_1. The arteries are segmented automatically using a modification of Frangi's vesselness filter [3, 6]. And next to obtain contrast agent transition rate between arteries and the ROI, the two curves are fitted with the following *compartment model* [2]:

$$\dot{q}_2(t) = d\,(q_1(t - t_d) - q_2(t))$$

where d is a transition parameter and t_d is a delay between the two curves (Figure 2). TMP perfusion levels are exponentially related to the parameter d.

3 Results

The proposed method was validated with sequences obtained from patients at the hospital Na Bulovce, Prague, Czech Republic. Selected sequences are 8 bit 480x480 pixels, containing 75–100 frames, 12.5 frames per second, produced by a single-plane X-ray angiographic machine Philips Integris H. In the first experiment, tracked ROIs were compared to manually defined ROIs. When initialized in a single frame the mean overlap between automatic and manual ROIs was $59 \pm 6\%$, significant improvement to $84 \pm 7\%$ can be achieved when initializing along a complete heart beat.

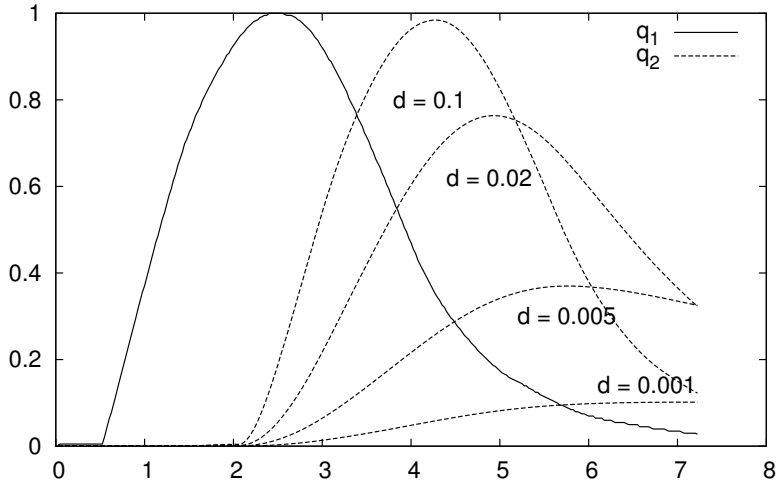


Figure 2: Compartment model with predicted ROI opacity curves for given arterial opacity curve and various transition parameter values.

The second experiment (Figure 3) shows that background subtraction significantly improves readability of the images. Using only matching frames in background estimation gives better results as we exclude poorly aligned frames, and thus a sharper background is obtained. Also using less frames accelerates the algorithm.

The third experiment shows results produced by the perfusion model applied to real data (Figure 4).

We implemented an interactive tool containing the described method in C++ using the ITK toolkit [5]. Registration is computed offline to permit interactive exploration.

4 Conclusions

We have proposed a method to enhance angiographic sequences by automatic alignment and background subtraction and to extract opacity curves for user-selected ROIs. This means easier and faster visual classification as well as it enables the proposed semi-automatic perfusion level estimation. As far as we know, there are no other methods that deal with coronary opacity quantification. Since registration is essential to our approach, a better image registration technique less affected by auxiliary objects (catheter, electrode) and objects not important for our application (ribcage) would improve the results.

Further details can be found in [6].

References

[1] D. S. Baim and W. Grossman, *Coronary angiography*, Philadelphia: Lippincott Williams & Wilkins, 2000.

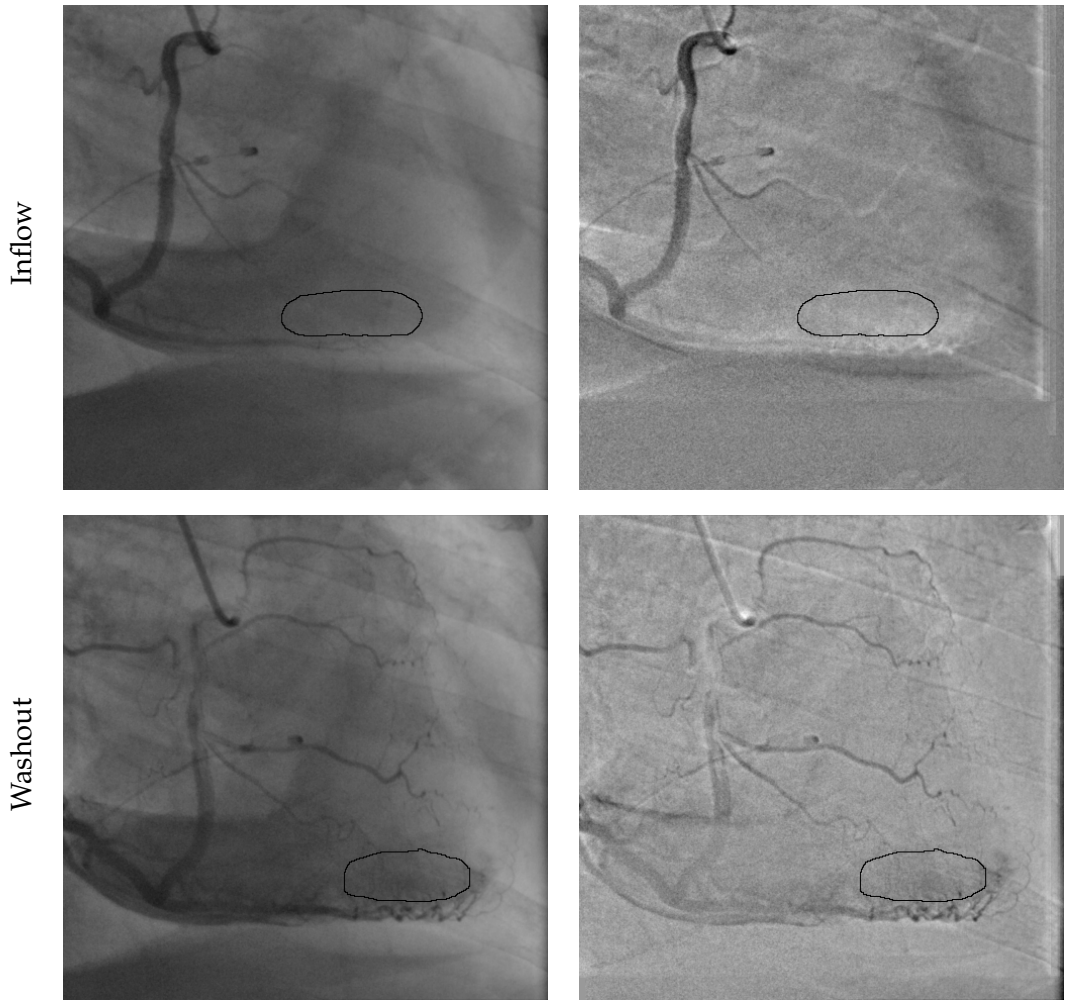


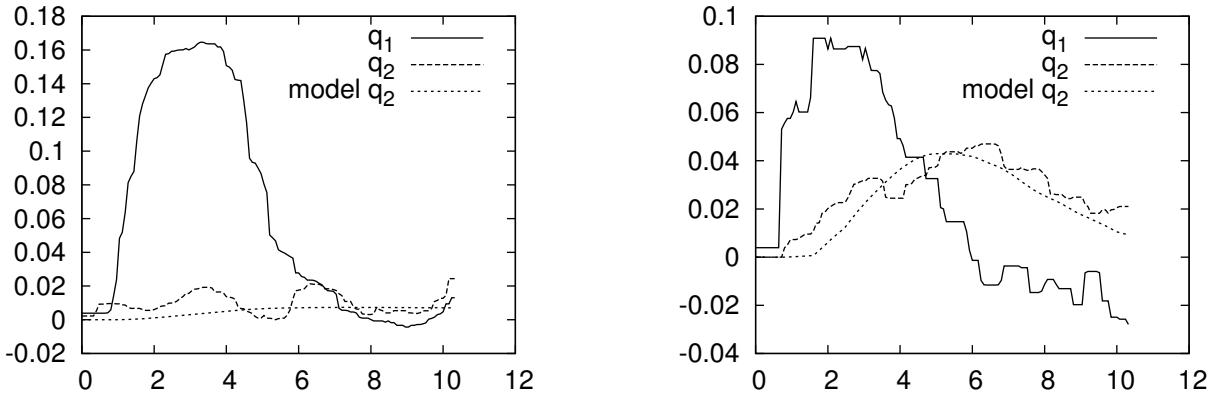
Figure 3: Background subtraction — compare how opacification is visible in the region next to an artery.

[2] M. Blomhøj, T.H. Kjeldsen, and J. Ottesen, *Compartment models*, 2005, Available at `http://www4.ncsu.edu/~msolufse/Compartmentmodels.pdf`.

[3] A.F. Frangi, W.J. Niessen, K.L. Vincken, and M.A. Viergever, *Multiscale vessel enhancement filtering*, Lecture Notes in Computer Science **1496** (1998), 130–138.

[4] C.M. Gibson, C.P. Cannon, S.A. Murphy, K.A. Ryan, R. Mesley, S.J. Marble, C.H. McCabe, F. Van de Werf, and E. Braunwald, *Relationship of TIMI myocardial perfusion grade to mortality after administration of thrombolytic drugs*, Circulation **101** (2000), 125–130.

[5] L. Ibáñez, W. Schroeder, L. Ng, and J. Cates, *The ITK software guide*, The Insight Software Consortium, 2005, Available at `http://www.itk.org/ItkSoftwareGuide.pdf`.



(a) $d = 0.001$, $t_d = 0$ (b) $d = 0.02$, $t_d = 0.8$

Figure 4: Opacity curves and the fitted model applied on data from two patients, in (a) the model does not detect any opacification, in (b) opacity persists a long time.

[6] T. Kazmar, *Opacity quantification in cardiac angiogram sequence*, Master's thesis, MFF-UK, Prague, Czech republic, 2008, Available at `http://tsh.plankton.tk/diplo/diplo.pdf`.

[7] J. Kybic, *Fast parametric elastic image registration*, IEEE Transactions on Image Processing **12** (2003), 1427–1442.

[8] M. Unser, *Splines: A perfect fit for signal and image processing*, IEEE Signal Processing Magazine **16** (1999), no. 6, 22–38.

Evolutionary System Identification

DI Dr. Stephan M. Winkler
Heuristic and Evolutionary Algorithms Laboratory
School of Informatics, Communications and Media
Upper Austrian University of Applied Sciences
Softwarepark 11, 4232 Hagenberg / Austria
E-mail: stephan.winkler@heuristiclab.com

June 18th, 2008

Genetic Programming and System Identification

Evolutionary computation is a subfield of computational intelligence that uses concepts inspired by natural evolution: Solutions to a given problem are represented by individuals of a population of solution candidates, and these individuals evolve iteratively by repeated selection of parents for producing new individuals. One of the most famous evolutionary techniques is the genetic algorithm, a global optimization technique using aspects inspired by evolutionary biology such as selection, recombination, mutation and inheritance. Genetic programming (GP) is an extension to the genetic algorithm that is able to automatically search for computer programs that solve given problems. The so-called GP cycle [3], shown in Figure 1, represents GP's iterative concept of repeatedly creating and testing programs in order to obtain programs that are able to solve the given problem situation.

In principle, system identification denotes the generation of mathematical models for systems based not on a priori knowledge, but rather on measured data; the result of a system identification algorithm consists in a mathematical description of the behavior of the analyzed system. In the context of our work we concentrate on system identification techniques based on genetic programming: Mathematical expressions are produced by an evolutionary process that uses the given measurement data; what we want to get in the end is a formula that fits the given data as well as possible. Since system models are identified by evolutionary processes, this field is also referred to as evolutionary system identification.

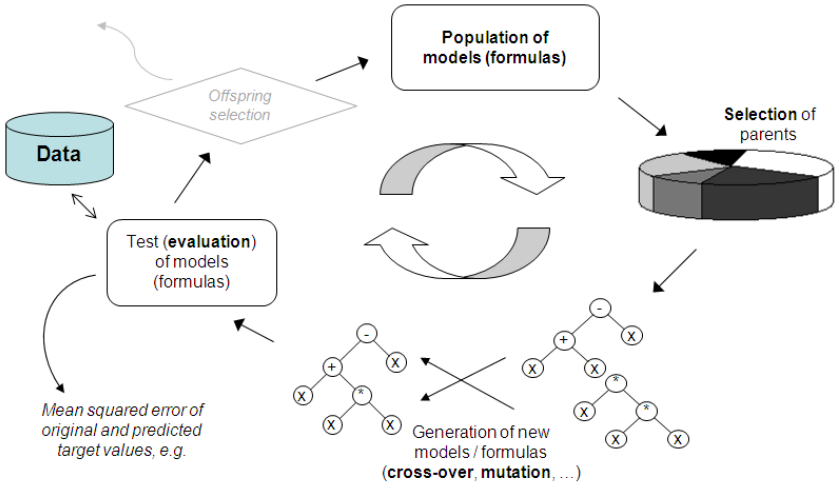


Figure 1: The genetic programming cycle.

In general, it is well known that every model is characterized by its formula structure and of values (parameters). System identification actually implies both, but usually the definition of the structure is considered either obvious or as the less critical issue, while the consistent estimation of the parameters especially in presence of noise receives the largest part of the attention. All this is quite understandable in the case of linear systems - the structure of a model exhibiting the same input/output behavior as the system under examination is in some sense at most a "multiple" of the actual structure - but is completely misleading for nonlinear systems. Therefore, while many scientists consider system identification as one of the most important research directions, in practice, however, the largest part of the nonlinear identification community has been working on theoretical questions or on the parameter identification of specific classes. For practical aspects, this boils down to setting up a list of candidate models and picking the best one - a procedure which is both time-consuming and hardly optimal. Using genetic programming for data based modeling brings along the advantage that we are able to design an identification process that automatically incorporates variables selection, structural identification and parameters optimization in one process. The function f which is searched for is not of any pre-specified form when applying genetic programming to data based modeling; during the GP process, low-level functions are combined to more complex formulas. Usually, standard arithmetical functions such as addition, subtraction, multiplication, and division are in the set of functions, but also trigonometric, logical, exponential, and other more complex functions can be included.

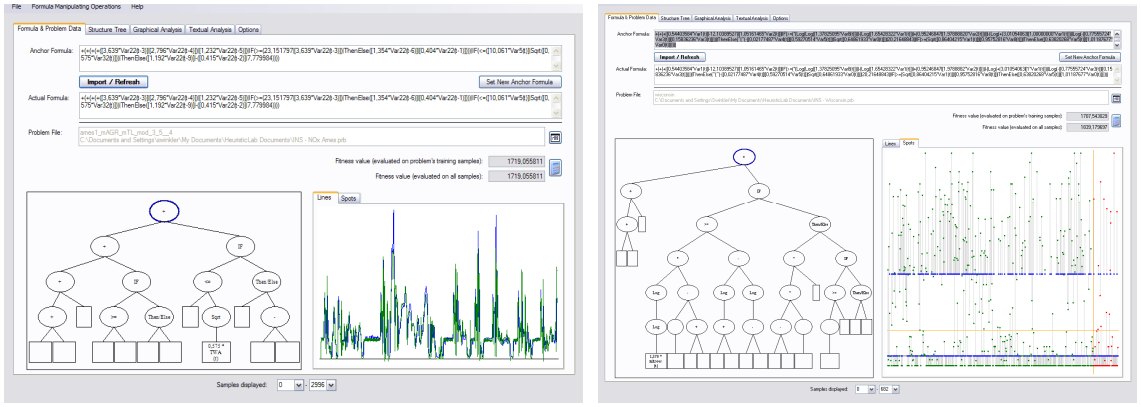


Figure 2: Left: Dynamic model for NOx exhaustions of a BMW diesel engine, right: Static classification model for a medical benchmark problem (Wisconsin data set, UCI repository).

Implementation of GP in HeuristicLab

Since 2002, members of the Heuristic and Evolutionary Algorithms Laboratory have been implementing the optimization framework HeuristicLab, a paradigm-independent and extensible environment for heuristic optimization. For this plugin-based framework a comprehensive implementation of GP based system identification has been developed; this implementation has been used extensively for testing algorithmic enhancements, especially concentrating on the following application domains:

- Analysis of nonlinear dynamic, mechatronical systems: In cooperation with the Institute for Design and Control of Mechatronical Systems at Johannes Kepler University Linz, Austria, we have intensively done research in the field of data based identification of models for the NOx and soot exhaustions of BMW diesel engines.
- Analysis of medical machine learning datasets: We have also intensively used our GP implementation for developing classifiers for machine learning problems, especially concentrating on medical classification problems in which the goal is to develop rules that can be used for classifying the health status of patients.

Figure 2 shows two labeled rooted structure trees representing mathematical models and their evaluation on a time series and a classification data set.

One of the most important factors that have enabled the high quality results documented in our articles surely is the use of strict offspring selection (OS). As is schematically shown in Figure 6, the main idea here is that individuals are (after being created and evaluated) compared to their own parents; they are designed to become a part of the next generation's population if and only if they are (in terms of solution quality) better than their parents. In [1], for example, basics and empirical results showing the effects of this enhanced selection scheme are described.

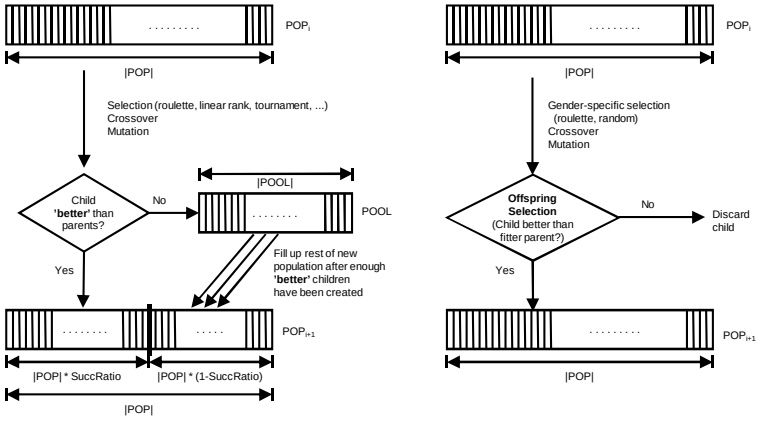


Figure 3: Offspring selection (left: original OS as published on [1], right: strict OS).

We have used our GP implementation for the HeuristicLab framework for identifying models for various domains; these applications include the following ones:

- Analysis of technical systems
  - Modeling the emissions of diesel engines
  - Prognosis of the quality of steel production products
- Analysis of bio-medical data collections
- Sales prognosis
- Generation of priority rules for production planning and optimization

In [6] a summary of respective research results in these application domains and references to respective publications can be found; for an overview of publications of the HEAL research group please see the HeuristicLab homepage[1].

Analysis of Population Dynamics in Genetic Programming

Of course, the use of extended selection concepts and other additional optimization phases (for parameter optimization or pruning) has strong effects on genetic processes and internal process dynamics. There have already been a lot of theoretical investigations of process dynamics in GP: After several ad-hoc engineering GP approaches which were developed and also published with great success in the 1990s, there was an increasing interest in how and why GP works. Even though GP was applied successfully for solving problems in various areas, the development of a GP theory was considered rather difficult even through the 1990s. Since the early 2000s it has finally been possible to establish a theory of GP showing a rapid development since then.

[1] http://www.heuristiclab.com/publications/index.html

Still, many of these theoretical considerations are only valid under very restricted circumstances, for example assuming no mutation, restricted function bases and very restricted terminal sets. This is why we have developed and applied a series of functions for monitoring internal process dynamics, and are thus able to demonstrate how the GP process is affected by additional selection or optimization stages. As documented in [6], some of these empirical process analysis approaches are: the analysis of genetic propagation, variables diversity and structural population diversity, e.g.

For example, similarity estimation functions can be used for estimating the structural similarity of solutions; thus we can estimate the diversity in populations and compare diversity dynamics in standard GP to diversity in enhanced GP implementations. We have compared standard GP with tournament selection to enhanced GP with gender specific parents selection and strict offspring selection.
For instance, we have used GP for identifying models for the NO_x emissions of a BMW diesel engine; standard GP with tournament selection ($k = 3$) as well as GP with random & roulette parents selection and strict offspring selection have been tested. Parts of the results of the analysis of these two GP strategies are shown in Figure 4: For both strategies the mutual similarities of solutions are displayed, the mean values being drawn as solid black lines.

- In standard GP the average mutual similarity of solution candidates (models) quickly increases and reaches values around 0.7, and then stays approximately in this level. (This is shown in the upper part of Figure 4).
- In the upper part of Figure 4 we exemplarily show this analysis for enhanced GP with random & roulette parents selection and strict OS: The mutual similarity of models steadily increases and finally reaches values of almost 1.0. Hereby we also see that the maximum possible value (1.0) is almost reached, when the selection pressure reaches the predefined limit – at this point the algorithm is no more able to produce offspring that outperforms its parents, and thus can be stopped.

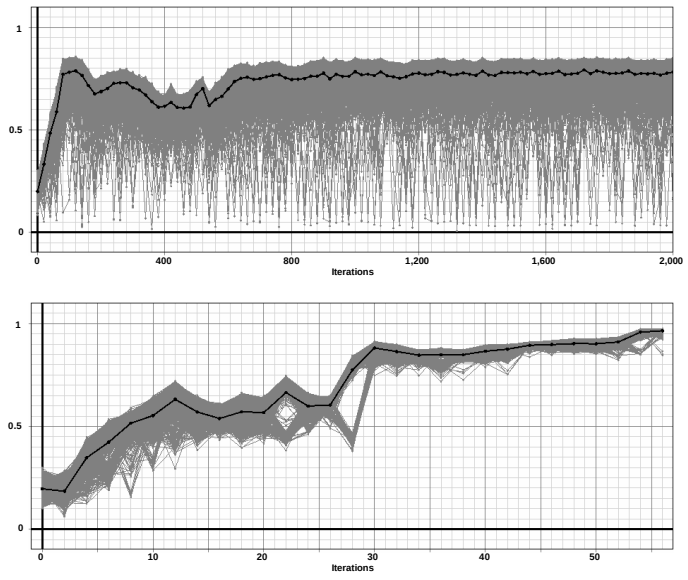


Figure 4: Single population diversity analysis based on the mutual similarity of solution candidates.

In the chats shown in Figure 4 we also see that the number of generations in GP with strict OS is a lot lower than in standard GP where normally the evolutionary process is executed for 1000, 2000 or even more iterations; in GP with strict OS the maximum selection pressure (normally set to a value as for example 200 or 500) is often reached after 100 or even less generations. Still, the effort needed for producing a new generation's population is increased significantly when applying OS, so that the overall effort (in terms of evaluated solutions or runtime consumption) is comparable for standard GP and GP with strict OS.

Acknowledgments

The work described in this paper was done within the Translational Research Project L284-N04 "GP-Based Techniques for the Design of Virtual Sensors" sponsored by the Austrian Science Fund (FWF).

References

[1] Michael Affenzeller, Stefan Wagner, and Stephan Winkler. Goal-Oriented Preservation of Essential Genetic Information by Offspring Selection. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2005*, vol. 2, pp. 1595–1596. Association for Computing Machinery (ACM), 2005.

[2] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* The MIT Press, 1992.

[3] William Langdon and Riccardo Poli. *Foundations of Genetic Programming.* Springer Verlag, Berlin Heidelberg New York, 2002.

[4] Lennart Ljung. *System Identification – Theory For the User, 2nd edition.* PTR Prentice Hall, Upper Saddle River, N.J., 1999.

[5] Stefan Wagner and Michael Affenzeller. HeuristicLab: A Generic and Extensible Optimization Environment. *Adaptive and Natural Computing Algorithms.* Springer Computer Science, pp. 538-541, 2005.

[6] Stephan Winkler. *Evolutionary System Identification - Modern Concepts and Practical Applications. THESIS.* PhD Thesis, Institute for Formal Models and Verification, Johannes Kepler University Linz, Austria, 2008.

Regularization for Diffusion Tensor Imaging

Leila Muresan
Fuzzy Logic Laboratorium Linz-Hagenberg
e-mail leila.muresan@jku.at

June, 2008.

Abstract

Our aim is to present a short survey on the Diffusion Tensor Imaging (DTI) technique, together with the challenges it implies from an image processing point of view, as well as potential applications of this imaging modality.

A brief explanation of the acquisition technique for magnetic resonance and diffusion weighted images is given.

We will see that the computation of the apparent diffusion coefficients (ADC) via the Stejskal-Tanner equations is sensitive to noise (especially in the low intensity range). Moreover, positive definiteness constraints have to be enforced.

Consequenlty, regularization of diffusion tensor fields is often required. In this presentation, we consider both approaches based on the diffusion weighted images and the tensor field. We will show that the latter case allows for ensuring the positive definiteness constraint either by projecting the solution at each iteration on the given subspace or by considering an appropriate (e.g. log-euclidian) manifold.

Further tasks (e.g. fiber tracing, registration, matching with existing atlases etc.) and several applications are concluding the talk.

GPGPU Approach for Parallelizing Support Vector Machines

Szilárd Páll

ISI Hagenberg

szilard.pall@isi-hagenberg.at

Master and PhD Seminar
SCCH, Hagenberg

June 18, 2008

Extended Abstract

In the golden age of parallel computing, during the 80s and early 90s and during the era of massively parallel machine based distributed computing the "parallel world" was a luxury. Nowadays parallel computing is becoming a widely accessible commodity technology. This process of democratization of parallel computing is greatly facilitated by the availability of hardware not explicitly tailored for parallelism but which is capable of massive parallelism, especially data-parallelism.

Graphical Processing Units (GPU-s) are designed for highly parallel, mathematically intensive computation. The scalar architecture designed for graphics is suitable for SIMD data-parallel computing. In the last five years the processing power of GPU-s has been dramatically increasing such that today graphics cards are at least an order of a magnitude faster than CPU-s. Programming GPU-s for general purpose computing has been already used for years but this was possible only with "tricking" the GPU-s to make data-parallel computations through rendering images.

The NVIDIA CUDA (Compute Unified Driver Architecture) is a GPGPU technology that provides programming interface using the C language for the last two generations of NVIDIA graphics hardware making parallel computing more accessible and bringing people closer to teraflops computing. With this framework, a regular desktop machine with CUDA capable graphics

cards as parallel mathematical coprocessor can be turned into a massively parallel machine capable of reaching the teraflops limit[1], which a couple of years ago was available only using traditional supercomputers.

Support Vector Machines is a supervised machine learning method for classification and regression. SVM is between the today's best performing method from the point of view of accuracy and generalization capability. The SVM algorithm carries the properties of parallelism. It operates on data from vector space, typically with a high dimensionality, which puts the data parallel GPU-s in a favorable position for speeding up the SVM algorithms.

In the frame of this thesis work the goal is to supplement the image processing framework of the SCCH company with GPGPU implementations, in particular the implementation of the SVM classification. Achieving speedup in the classification phase is especially important because this is the one of the two parts (training, classification), that is generally executed on-line. This means that most of the times the SVM classification is a time-critical operation that needs to be as quick as possible the enable flawless operation of the application that is integrated into. Therefor in the frame of this thesis a concept is developed for the SIMD parallelization of SMV decision functions and adapted to the NVIDIA CUDA programming model. An implementation using the aforementioned framework is made to demonstrate the concept and experiment with the possibilities of GPGPU implementation. Testing is made on real-life data from the field of texture analysis, provided by the company.

The results of this work show, that the great potential of GPGPU and the CUDA framework is suitable and useful for SVM parallelization. The current implementation, while it's not finished yet, but shows up to 15 times speedup compared to the single processor implementation. Although this is not consistent and for some test data it is even below 1. This is because the performance of the algorithm is highly dependent on dimensionality of the input data and the saturation of the arithmetical operations in the used method. Therefor it is hard to provide a general purpose implementation for wide range of applications but it is easy to tune the implementation for a specific application.

While the programming of GPUs needs adaptation to this programming concept and the methods and algorithms have to be deeply rethought to be massively parallel the benefits are remarkable and sometimes spectacular.

This ongoing masters thesis is done in the frame of the Intenational Masters Program Hagenberg, Austria with the support of the Software Competence Center Hagenberg.

[1]The G80 architecture is capable of 350 Gflops peak performance, while the G90 reaches 500 Gflops. With two G90 cards the 1 TFlops peak power is reachable.