ALBERT-LUDWIGS-UNIVERSITÄT
FREIBURG
INSTITUT FÜR INFORMATIK

Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr. Hans Burkhardt

# Segmentation of Biological Structures

in

# 3D Volumetric Data

using

# associative Markov Networks

## Diplomarbeit

Julia Wicklein

Juni 2006 – Dezember 2006

# Erklärung

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbstständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

Julia Wicklein

Freiburg, den 20.12.2006

# Danksagung

Ich bedanke mich sehr herzlich bei ***Prof. Dr.-Ing. H. Burkhardt***, der es mir ermöglicht hat, diese Arbeit am Lehrstuhl für Mustererkennung und Bildverarbeitung anzufertigen.

Ein besonderer Dank gilt meinem Betreuer ***Dipl.-Inf. Janis Fehr***. Dafür, dass er mir jederzeit mit Rat und Tat zur Seite stand und ohne den diese Arbeit so nicht möglich gewesen wäre.

Ich danke ***Margret Keuper*** und ***Maja Temerinac*** für die wunderbare moralische Unterstützung während dieser sechs Monate.

Ganz besonders möchte ich mich an dieser Stelle einmal bei meinem Vater ***Toni Eckert*** bedanken. Er hat immer an mich geglaubt und mich zu jeder Zeit meines Lebens aus vollstem Herzen unterstützt.

Meinem Ehemann ***Markus Wicklein*** danke ich dafür, dass er immer für mich da ist. Er gab mir das nötige Selbstvertrauen und die Willensstärke, für dieses Studium und darüber hinaus.

December 19, 2006

**Aufgabenstellung für die Diplomarbeit**
**von Frau Julia Wicklein**

**Segmentierung biologischer Strukturen in 3D Volumendaten**
**mit assoziativen Markov-Netzwerken**

Die Segmentierung biologischer Strukturen in 3D Volumendaten (z.B. Aufnahmen von konfokalen Laser Scanning Mikroskopen) ist ein äußerst schwieriges Problem. Bisher gelingt es mit Hilfe voxel-weiser Methoden [3], Objekte erfolgreich zu segmentieren und zu klassifizieren. Allerdings ist es mit diesem Ansatz nicht möglich, sich berührende Objekte zu trennen, wenn diese der selben Klasse angehören. Zudem kann es zu lokalen Fehlklassifikationen kommen. In einem zweiten Schritt sollen nun in dieser Arbeit mit Hilfe von assoziativen Markov-Netzwerken [1][2] die Ergebnisse aus [3] weiter verarbeitet werden, um die genannten Probleme zu beseitigen:

- Entwurf und Implementation geeigneter Markov-Netze zur Segmentierung künstlich erzeugter 2D und 3D "Toy"-Daten.
- Entwurf und Implementation geeigneter Markov-Netze zur Segmentierung biologischer Volumendaten.
- Entwurf und Implementation eines Lernalgorithmus zur automatischen Generierung von Markov-Netzen anhand von Trainingsdaten [2].

Literatur:


[1] Anguelov, D. et. Al: *Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data*, CVPR 2005

[2] Taskar B., Chatalbashev V., Koller D.: *Learning associative Markov Networks*, ICML 2004

[3] Fehr J, Ronneberger O., Kurz H. Burkhardt H.:*Self-learning Segmentation and Classification of Cell-Nuclei in 3D Volumetric Data using Voxel-Wise Gray Scale Invariants*, W. Kropatsch, R. Sablatnig, A. Hanbury (Eds): DAGM 2005, LNCS 3663, pp 377-384, 2005


| | |
|---|---|
| *Referent:* | Prof. Dr.-Ing. H. Burkhardt |
| *Betreuer:* | Dipl.-Inf. J.Fehr |
| *Ausgabedatum:* | 21. 06. 2006 |
| *Abgabedatum:* | 20. 12. 2006 |
| *Bearbeitungszeit:* | 6 Monate |

# Abstract

This thesis introduces associative Markov networks for the segmentation of biological structures in 3D volumetric data. It is built upon the basis set by the work in [4], where the segmentation and classification of objects using voxel-wise gray scale invariants is presented. This work makes further refinements to [4] which is of interest, because objects which lie close together, are sometimes misclassified as one.
Markov chain Monte Carlo sampling is introduced and an appropriate learning algorithm is presented for solving the optimization problem.
For feature extraction, a new method is presented, that calculates fast local curvature estimation. It shows satisfying results and is robust towards noise. A smoothing algorithm has been implemented that removes small fractions of noise. A futher extension of this work supplies the possibility to split overlapping objects at a marked position.
All implemented methods show very promising results on 3D volumetric samples of chicken embryo chorioallantoic membrane (CAM) probes, recorded with a confocal laser scanning microscope.



# Zusammenfassung

In dieser Diplomarbeit werden assoziative Markov Netzwerke zur Segmentierung von biologischen 3D Volumendaten präsentiert. Sie baut auf der Arbeit [4] auf, in der Segmentierung und Klassifikation von Objekten mit Hilfe voxelweiser Grauwertinvarianten vorgestellt werden. Diese Arbeit verbessert die Methoden aus [4], was von Interesse ist, da nahe beieinander liegende Objekte manchmal fälschlicherweise als eines klassifiziert werden.
Markov-Ketten Monte Carlo Sampling sowie ein geeigneter Lernalgorithmus zur Lösung des Optimierungsproblems werden vorgestellt. Zur Merkmalsextration wird eine Methode vorgelegt, die eine schnelle Schätzung der lokalen Krümmung vornimmt. Sie zeigt zufriedenstellende Ergebnisse und ist robust gegenüber Rauschen. Ein Glättungsalgorithmus, der schwaches Rauschen unterdrückt, wurde implementiert. Ein weiterer Teil dieser Arbeit erlaubt es, sich überlappende Objekte an bestimmten Positionen voneinander zu trennen.
Alle implementierten Methoden zeigen vielvesprechende Ergebnisse auf 3D Volumendaten von Gewebeproben aus der Chorioallantois-Membran des Hühnereis, die mit einem konfokalen Lasermikroskop aufgenommen wurden.

# Contents

# 1 Introduction

The topic of this work runs as: “segmentation of biological structures in 3D volumetric data using associative Markov networks”. Segmentation of structures in 3D datasets is a very difficult task. However, a satisfying solution to that problem would bring a quite long list of benefits along with it.
The data that has been made disposable for this work includes 3D volumetric samples of chicken embryo chorioallantoic membrane (CAM) probes, recorded with a confocal laser scanning microscope.[1] CAM is a suitable model for some very promising medical research areas. Especially the developement and adaptation of new bloodvessels, the so-called *Angiogenesis* is in the focus of many explorations. New perceptions within this research area would progress the understanding of many frequent diseases, including cancer and heart ischemia.
Regarding a solid tumor for example: This is much more complex than the simple pictorial idea of a collection of increasing tumor cells. It rather includes the tumor cells on the one and the tumor stroma on the other hand. This stroma consists of connective tissue and bloodvessels. Its growth offers the tumor supply through the upleading bloodvessels. Therefore, cells building the vessel walls are important parts of recent research. Common tumor therapy methods focus on killing current tumor cells and trying to prevent their further propagation. This leads to a number of problems and negative side effects on chemo- or radiation therapies. Angiogenetic therapies instead are nearly free of side effects. Especially an automated localization and classification of different cell types in 3D tissue probes would lead to a drastical reduction of the time spent by human experts.

In [3], an algorithm is presented that uses voxel-wise gray scale invariants for segmentation and classification of objects. This works quite well for the database and is introduced in Chapter 2, but there are still problems left. This approach has no appropriate solution if an object of one class touches another object of the same class. Also it is possible that local missclassifications happen.
The main task for this thesis is to find appropriate solutions to these problems, using associative Markov networks and different types of feature extraction methods.

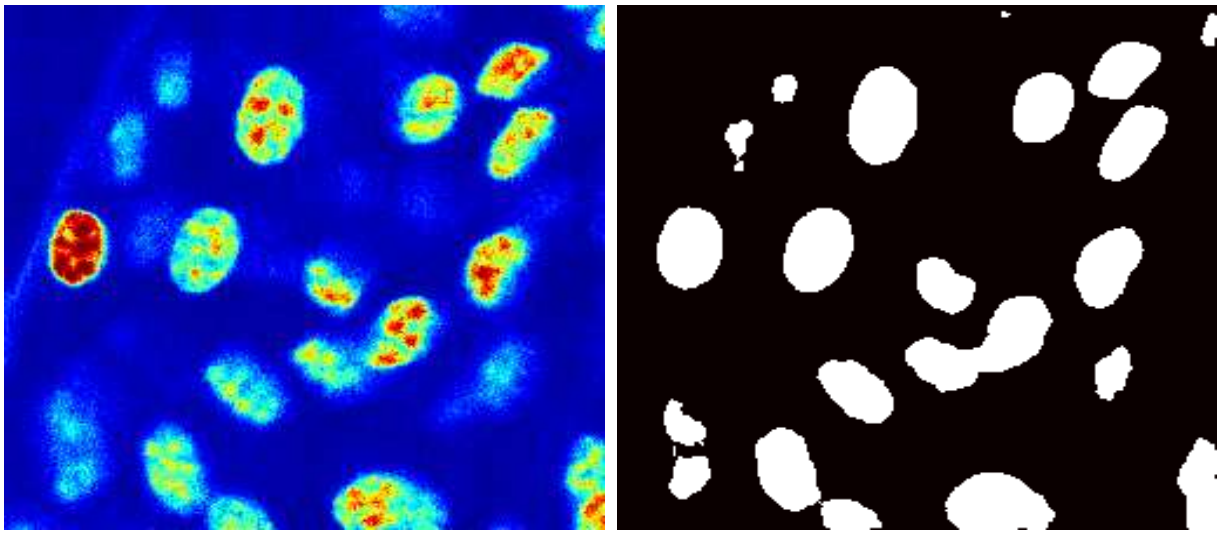


Figure 1: Left: Slice of a 3D database entry; Right: The binary classification result from [4].

[1] http://anna.anatomie.uni-freiburg.de/kurz/.

## Structure of this work

Giving a comprehensive view on the structure of this work, Figure 2 is presented. It shows all particular steps towards the aspired goals of this thesis. The next two chapters treat with the upper bounding-box of the illustration. They review the work from [4] and exhibit problematic aspects. Chapter 2 gives an overview on the retrieval of the data used for this work, including fluorescence marking and confocal laser scanning microscopy, followed by a description of the database itself. In Chapter 3, an introduction to voxel-wise gray-scale invariants is presented and explanations on segmentation and classification methods that had been applied in [4] are given.
Afterwards, the second part of Figure 2 is explained that gives proper solutions to the mentioned problems. Chapter 4 treats the basic functionality of associative Markov networks, Markov chain Monte Carlo sampling and the resulting learning algorithm. The different types of feature extraction methods are introduced in Chapter 5, including a new approach for fast local curvature estimations. A toy dataset is shown that has been chosen for better illustrations of the features calculation steps and results. The last chapter presents the conclusions and an outlook for further research possibilities.

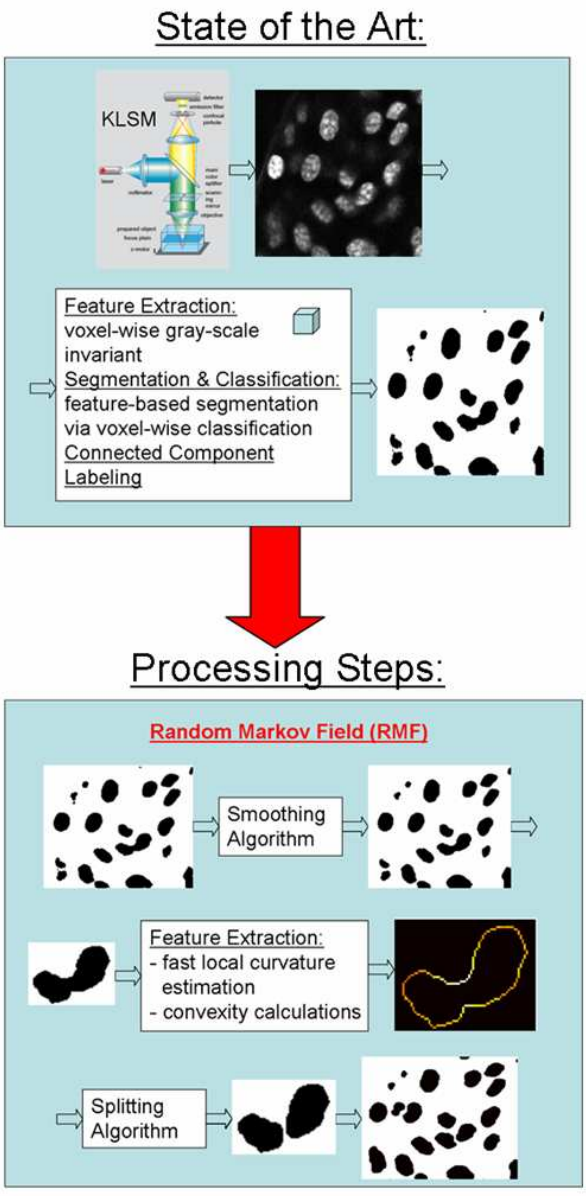



Figure 2: Overview on this work.

# 2 Data

## 2.1 Data Retrieval

### 2.1.1 Fluorescence Marking

*"Light beeing absorbed by a fluorescent substance has a smaller electro-magnetic wavelength than the light that is emitted afterwards."* Strokes,1852

This rule describes the main idea behind the principle of fluorescence microscopy. In this case, certain electrons of the fluorescenting molecules of the substance absorb the incoming photons and therefore reach a higher energy-level. The electrons on the other hand cannot hold on to that and drop immediately back to the original energy-level. During this process, the received energy is set free again and as one part of it, the so-called fluorescenting-light is emitted. This kind of light has a lower energy-level than the stimulating light and thus a larger wavelength. The average difference between stimulating and emitted lights' wavelengths lies about 20 to 40*nm*. Figure 3 shows these differences for two fluorescenting substances, Cy-3 and Yo-Pro:

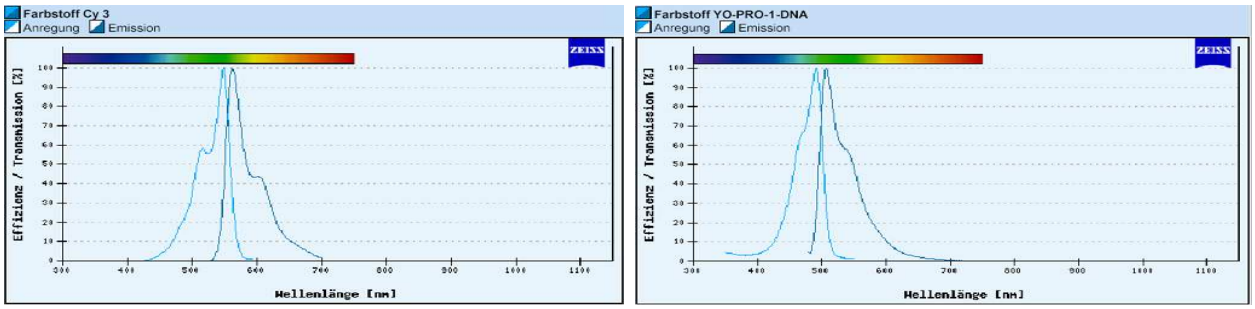



Figure 3: Wavelengths of two fluorescent substances [5].

Fluorescenting microscopes use exactly these properties. Through a suitable choice for the right filter inside the microscope, the brighter stimulating light could be spread away from the weaker fluorescenting light inside the pencil. So only those parts of the probe remain, that were previously prepared with a certain fluorescenting marker.

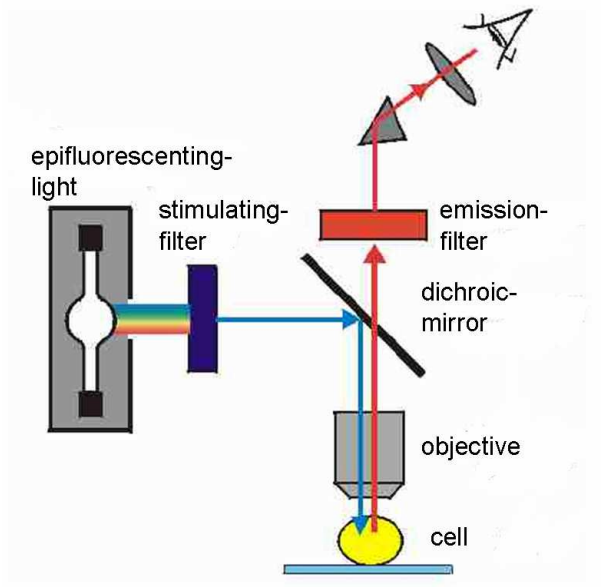



Figure 4: Diagram of a fluorescenting microscope [7].

### 2.1.2 Confocal Laser Scanning Microscopy

In order to avoid interferences like background noise or crosstalk and to improve focusing on the essential parts of the probe, a laser should be used as lightsource for the microscope. The big advantage is, that a fixed wavelength is emitted which can easily be focused with also higher resolution. By using several laser-lightsources, one single probe could be provided with multiple fluorescenting substances, and afterwards recorded into different channels. The structure of such a confocal laser scanning microscope (LSM) is illustrated in Figure 5:

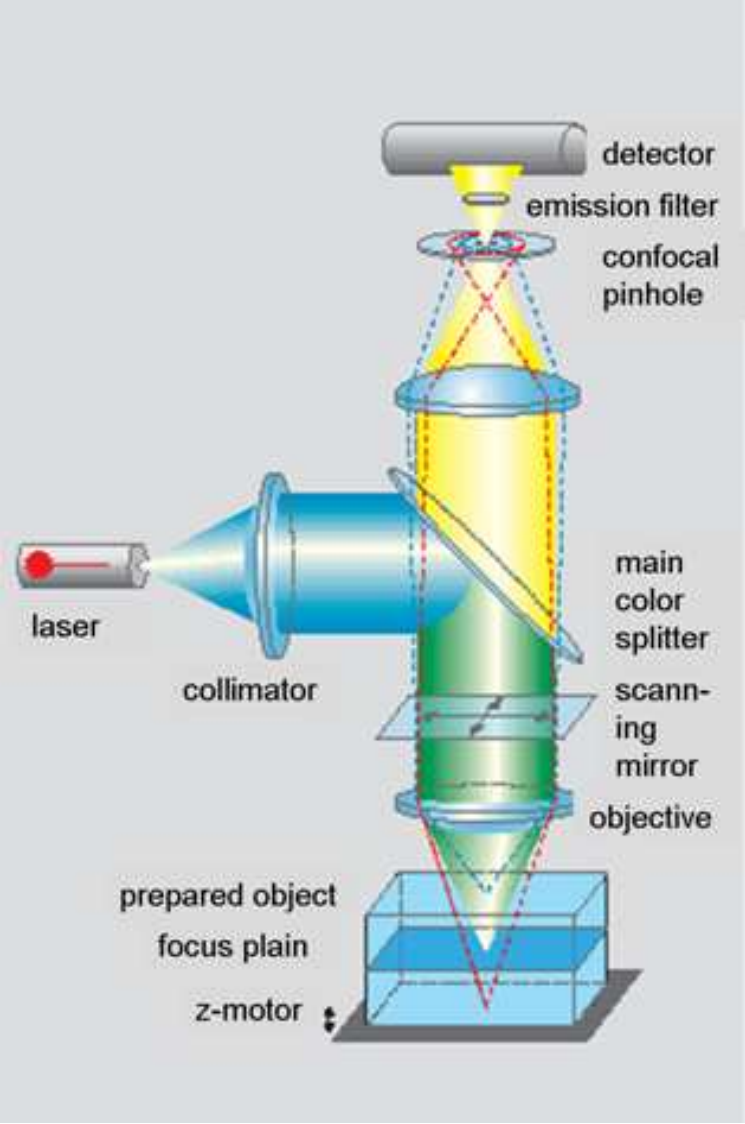



Figure 5: Confocal laser scanning microscope [5].

One part of the microscope is the so-called confocal pinhole. This makes it possible to blind out the outerfocal object informations and its diameter determines the thickness of the optical section. The figure shows, that only those paths, that lie in focus can pass the pinhole. Those marked red or blue are outside the focal plane.
The probe can now easily be scanned slice by slice with a certain z-resolution and three-dimensional datasets of the fluorescent marked structures are generated. The quality of the data depends therefore on the distance between two optical sections.
But there is also one problem left. The z-resolution is still not as good as the resolution achieved in x- and y-direction.

## 2.2 Cell Database

The database consists of 3D volumetric data samples of chicken embryo chorioallantoic membrane (CAM) probes. CAM is a common model for angiogenesis research at cellular level. It gives excellent conditions for the exploration of angiogenesis mechanisms, growth factors or their receptors. The data samples had been prepared with YoPro-1 and with Cy3 or Alexa546 fluorescent markers. After that, two different channels with YoPro(509nm) and Cy3/Alexa546(570nm) were recorded using a confocal laser scanning microscope (LSM) as described above.

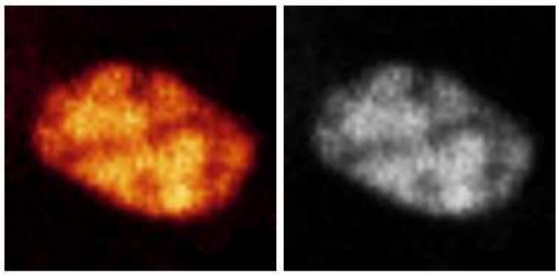


Figure 6: A LSM recorded erythrocyte nucleus marked with YoPro. Left: Pseudo coloration; Right: Gray scale.

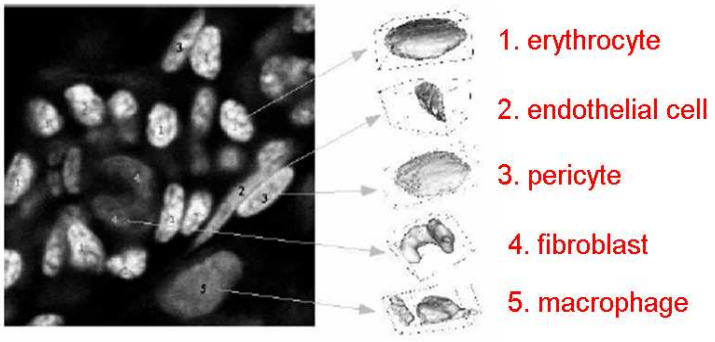



Figure 7: YoPro channel of the data, showing cross section of a capillary and a 3D reconstruction of the different cell types.

Figure 8: Slice of a sample 3D database entry (erythrocyte). Left: YoPro stained channel; Center: SNAAlexa stained channel; Right: Ground truth segmentation and label.

# 3 State of the Art

Having a look on Figure 2, this chapter introduces the second part of the upper bounding-box. All methods that are explained as state of the art have already been applied in [4]. First of all, voxel-wise gray-scale invariant features are presented that have been extracted from the database introduced in Chapter 2. After that, segmentation and classification is illustrated, followed by the results of these applications. The last section of this chapter exhibits the problems that are still left and which should by solved during this work.

## 3.1 Voxel-Wise Gray Scale Invariants

Feature-based methods, combined with learning algorithms have been found out to be a successful tool for image analysis also within biomedical research problems. This is true especially for those applications that aid the human experts fulfilling the final image analysing steps. One big challenge for computer aided systems lies in the group of classification problems. In the following, voxel-wise gray scale invariant features are explained that show very good results for classification tasks with 3D volumetric data samples in medical and biological applications.
The first step towards extracting an adequate feature for a certain problem is to find some function $T(X_i)$ that maps all input signals $X_i$ belonging to the same semantic class onto one single point $\tilde{x}_i$ inside the given feature space. Considering a concrete number of possible variations like 3D rotations drastically reduces the amount of training examples for the later classification step. All selected transformation possibibilties are put together into one transformation group $G$. After all elements $g$ of $G$ are known, the feature function $T$ can be constructed. The following equation illustrates the invariance of $T$ towards all transformation group entries $g$:

$$T(gX_i) = \tilde{x}_i, \forall g \in G. \tag{1}$$

Voxel-wise gray scale invariant features are generated for each single voxel in a given data volume by calculating Haar-intergration over the whole transformation group, meaning over all degrees of freedom of $G$:

$$T(X) := \int_G \big(gX\big)\mathrm{d}g. \tag{2}$$

To improve the often weak separability properties of features, non-linear kernel functions should be embedded into the integral. This can be achieved for an n-dimensional dataset $X$ in the following way:

$$T[f](X) := \int_G f\big(gX\big)\mathrm{d}g, \tag{3}$$

where $f$ denotes a nonlinear kernel function and $gX$ the transformed n-dimensional dataset.

The kernel function $f(X)$ can be rewritten as $f\big(X(x_1), X(x_2), X(x_3), \cdots\big)$ if it only depends on a fixed number of points of the volume, where $X(x_i)$ describes the gray value of the $i$th position. Therefore, we can now rewrite equation (3) as:

$$T[f](X) := \int_G f\big(X(s_g(x_1)), X(s_g(x_2)), X(s_g(x_3)), \cdots\big)\mathrm{d}g, \tag{4}$$

where the transformation of the kernel point $x_i$ is given by $s_g(x_i)$.
In general, kernel functions operate on scalar values. Most imaging devices return intensity values interpreted as gray-values, kernels are often referred as gray-scale kernel functions.
In order to achieve applicable group-integral features for large 3D datasets, a fast calculation method is needed.

The so-called seperable two-point gray-scale kernels offer a possible solution to that problem for Euclidean transformations by using fast convolution via FFT. They consist of the following form:

$$f(X) = f_a\big(X(0)\big) \cdot f_b\big(X(q)\big) \tag{5}$$

where $f_a, f_b$ are simple nonlinear functions transforming the gray values, for example: $f(x) = x^2, x^3, \cdots, \sqrt{x}$ etc.; $q$ is representing the span of the kernel function $f$.
One major drawback of the voxel-wise calculation with two-point kernel functions is the fact that two-point kernels are not only invariant towards rotation, but also to any random permutation of neighboring gray-values.
Figure 9 illustrates this dilemma. It is rather necessary to restrict invariance properties of features to transformations which preserve the intrinsic information of the classes.

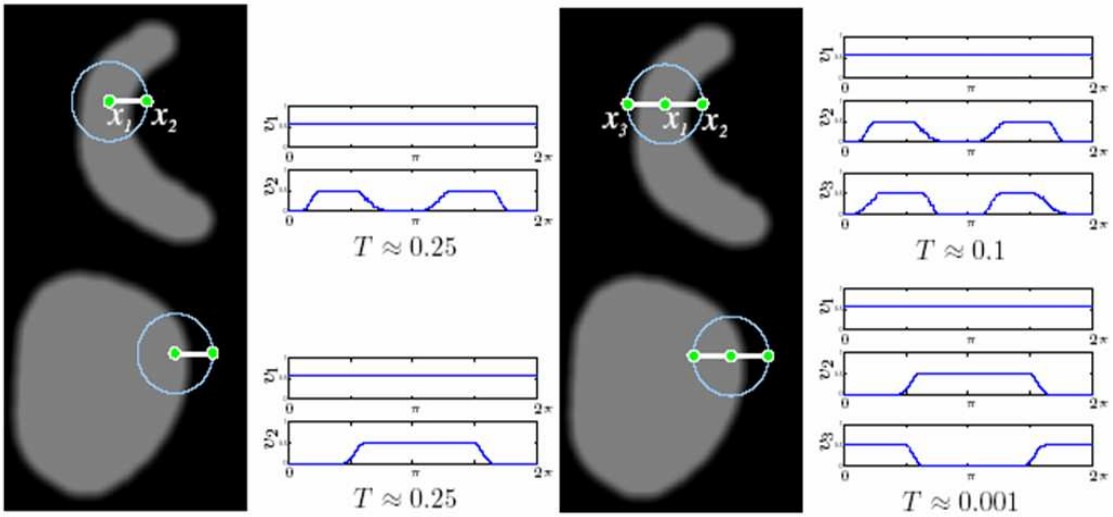



Figure 9: Local feature using the two-point kernel function (left) and voxel-wise features using three-point kernel functions (right) [8].

The solution to that problem is also presented in Figure 9. It is achieved by adding a third point to the support of the two-point kernel functions:

$$f(X) = f_a\big(X(0)\big) \cdot f_b\big(X(q_2)\big) \cdot f_c\big(X(q_3)\big) \tag{6}$$

Like for two-point kernels, the first point is located at the rotation center. These three-point-kernels are not sensitive to such permutations, but unlike the two-point variant, they cannot be calculated just in form of one simple convolution. However, it is possible to approximate them by an expansion in Spherical Harmonics, which leads to a series of simple convolutions, where every truncated evaluation of this series still fulfills the invariance criterion.

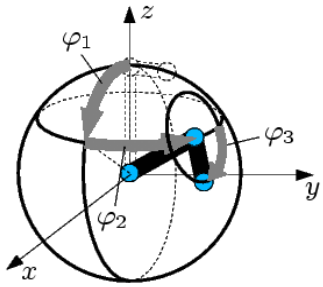



Figure 10: Parameterization of the 3D rotation with $\lambda = (\varphi_1, \varphi_2, \varphi_3)$.

Figure 11 briefly illustrates the computation instructions for the three-point kernel-functions. More detailed informations about designing features for 3D volumetric data in biomedical image analysis can be found in [8].

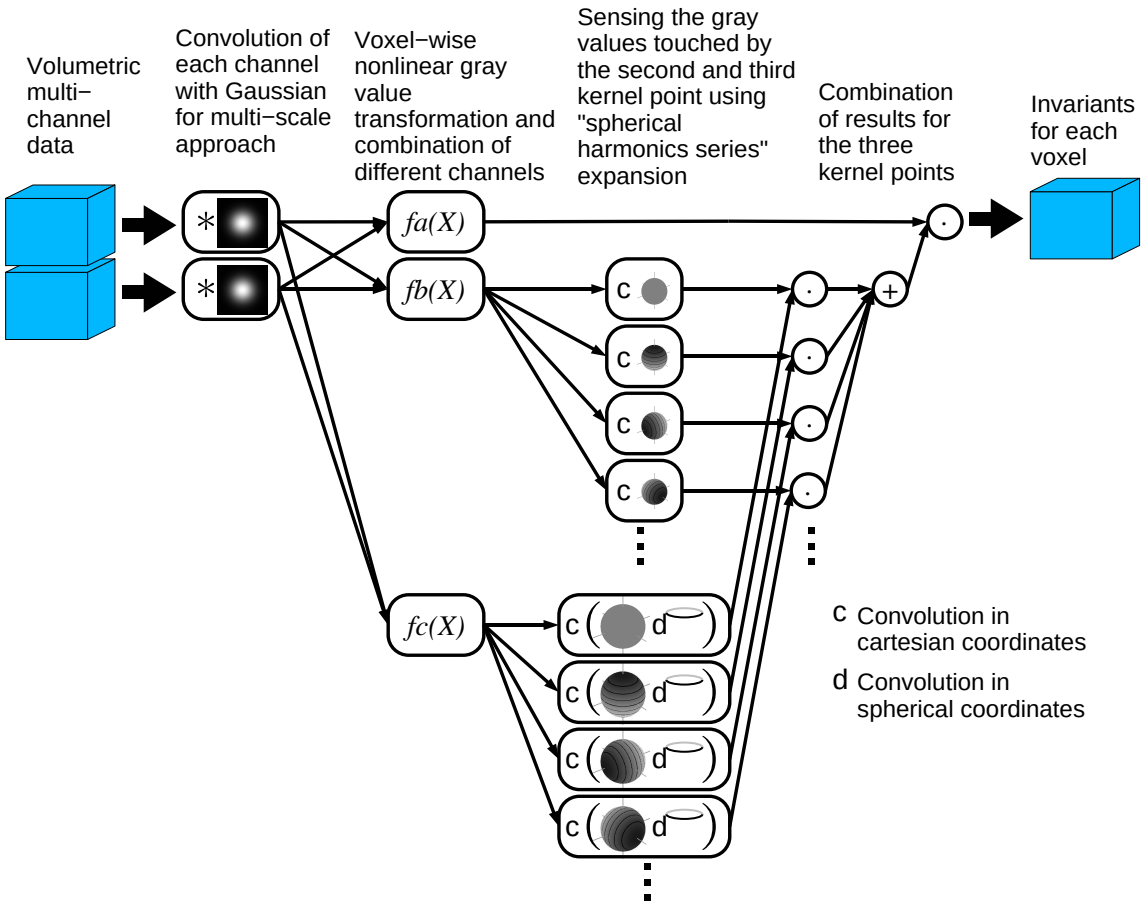



Figure 11: Computation of three-point-kernels $f(X) = f_a\big(X(0)\big) \cdot f_b\big(X(q_2)\big) \cdot f_c\big(X(q_3)\big)$ on multi-channel 3D volumetric data. For each kernel function this scheme simultaniously calculates the features for all voxels.

## 3.2 Segmentation and Classification

In general, the order for achieving a classified object out of a dataset would be segmentation, feature extraction, followed by classification. But this procedure has one major drawback for this approach. Segmentation of complex objects from datasets with structured background as it is given here, would require higher semantic a priori knowledge about the objects in order to identify the regions of interest. This would lead to very specialized model driven solutions, which cannot easily be adopted to new data or other structures.
To overcome this problem, a self-learning algorithm is applied, merging segmentation and classification into one single step inside the processing pipeline. The first step is now voxel-wise feature extraction, followed by feature-based segmentation via voxel-wise classification.
After the voxel-wise extraction of suitable feature-vectors using two- and three-point-kernels, a support-vector machine model is trained in an interactive procedure over several iterations. Therefore, a small number of training samples (voxels) for each class is manually selected by a human expert. At last, all voxels are classified against the previously trained model. New training samples can be added in each iteration to improve classification accuracy until a “stable” state is reached.
Finally, the labeled voxels could be combined to closed objects using connected component labeling.

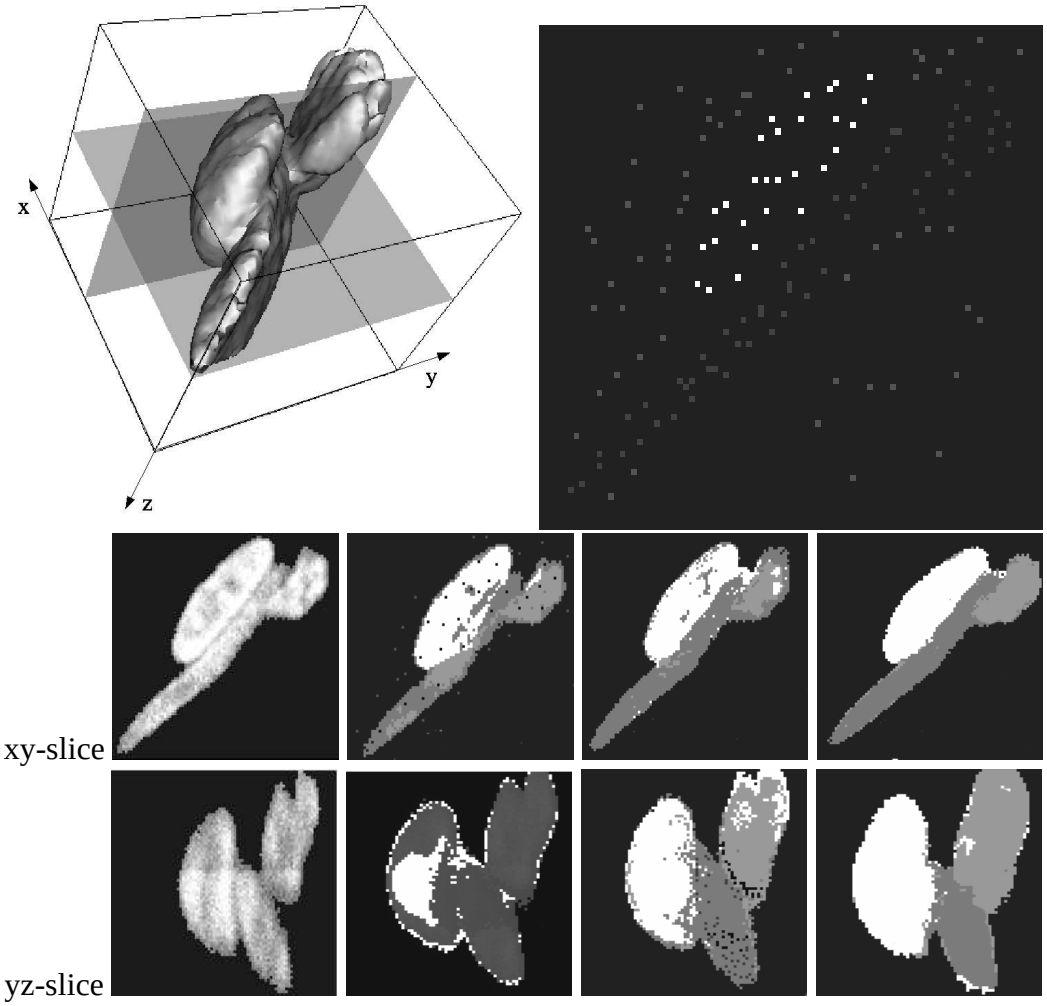



Figure 12: The interactive training process - 1st row: 3D reconstruction of the original data, 311 training samples set for the first iteration of training. 2nd line - from left to right: Section of xy-slice of original data as indicated in the 3D reconstruction, result after the first iteration (56 support vectors in model), result after 2nd iteration, result after the 3rd iteration (642 training samples, 129 support vectors in model). 3rd line: Section of yz-slice of original data, results after 1st to 3rd iteration in yz-slice.

## 3.3 Results

The experiments had been executed using the leave-one-out method. This means that all feature-vectors of one dataset-entry were taken away before training the SVM with the other entries. After that, the one that had been taken away was classified against them.

| cell type | recognition-rate/voxel | recognition-rate/cell |
|---|---|---|
| erythrocytes | 90,4% | 95,3% |
| endothelial cells | 81,2% | 84,6% |
| fibroblasts | 78,3% | 79,8% |
| background | 94,1% | - |

Table 1: Results of the method introduced in [4].

## 3.4 Problems

The algorithm introduced above is able to automatically detect previously learned objects. Even low fluorescent activity and strong intracellular structures do not cause false or partial segmentation results.
One problem has already been mentioned in Chapter 2.1.2. during the explanation of LSMs. The low z-resolution of the 3D datasets can cause missclassifications at object borders and noise. Another drawback arises with the rather simple approach of connected component labeling, as it is not capable of suppressing small fractions of noise. It is also not possible to split touching objects of the same class with this procedure.

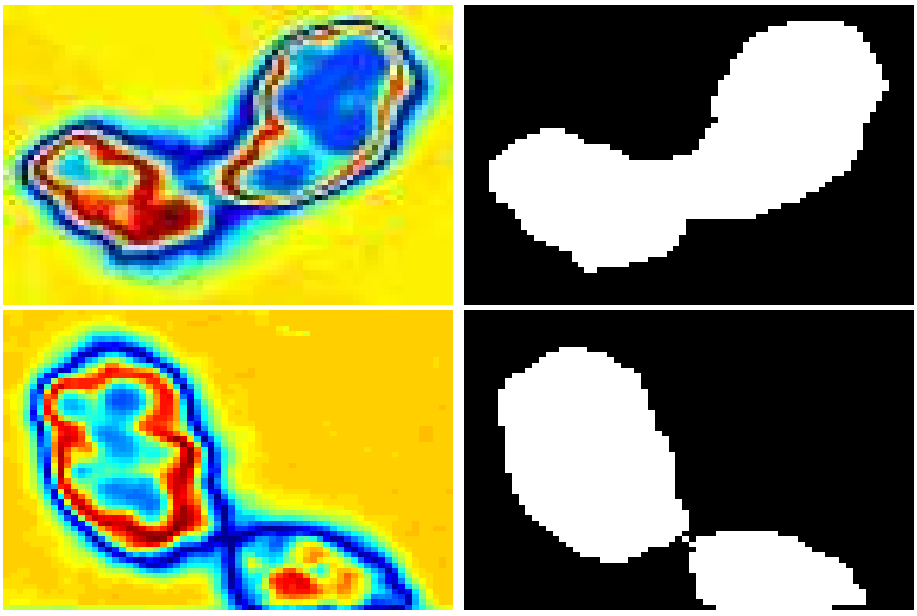


Figure 13: Left: Classification confidence; Right: Binary classified object.

Figure 13 illustrates the problem caused by two objects that are lying too close to each other. After the classification is fulfilled, they would be counted as one single object inside the dataset. Both images on the left side of the illustration show the distances from the seperating hyperplane inside the support vector machine used for classification. This can be interpreted as the confidence in selecting the labeling value. The blue regions have a weaker certainty than the red ones.

One can assume that choosing a threshold would be a good solution to the problem of seperating touching objects. The problematic areas are in fact of weak confidence. The picture illustrates, that there are also regions inside the objects that are marked with blue. It becomes clear that this approach would lead to undesired holes inside the objects.

# 4 Associative Markov Networks

With this chapter, the second part of Figure 2 is reached. It starts with a describtion of associative Markov networks. Markov networks are undirected graphical models that can appropriately describe all kinds of dependencies between different variables. The specialization to associative Markov networks is used for this work to give a suitable environment for applying optimized feature extraction methods. In this chapter, Markov networks are explained, followed by a describtion of Markov Chain Monte Carlo Methods (MCMCM). These methods, are a class of algorithms for sampling from probability distributions based on constructing a Markov chain.

## 4.1 Markov Networks

In the following, Markov networks over discrete variables $Y = \{Y_1, \cdots, Y_N\}$ are introduced. Each variable represents one single object that could be classified to one of $K$ possible labels, so that: $Y_i \in \{1, \cdots, K\}$. With N different objects, a markov network for $Y$ defines a joint distribution over $\{1, \cdots, K\}^N$. $y$ describes one possible assignment of values to $Y$. As already mentioned, Markov networks (or random fields MRFs) are defining an undirected graph $G(N, E)$. The set of nodes $N$ represents the object-variables. These are arranged in different groups, the so-called cliques, which form fully connected subgraphs. Two nodes $i$, $j$ are connected if there is an edge $(i, j) \in E$ from node $i$ to node $j$. Some possible neighborhoods for 2D and 3D are illustrated in Figure 14. Other variants are also considerable.

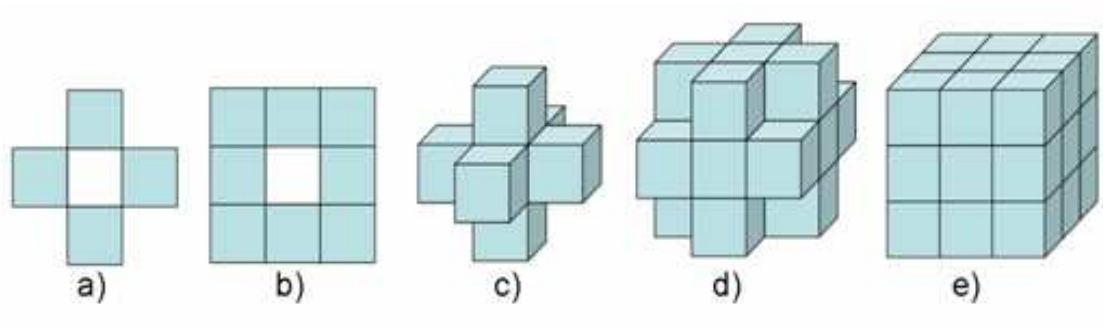



Figure 14: 4- and 8-adjacent pixels a) and b); 6-, 18- and 26-adjacent voxels c), d) and e).

To make it more evident, only pairwise Markov networks are regarded, meaning that all cliques consist of two nodes and one edge between them. Nodes and edges are associated with so-called potentials $\phi_i(Y_i)$ and $\phi_{ij}(Y_i, Y_j)$, with: $ij \in E(i < j)$. These potentials attache a non-negative value to each node, represented by the variable $Y_i$, respectively to each edge, represented by a pair of variables $Y_i, Y_j$. Litterally speaking, the potentials reflect how well the features fit to the labels. The joint distribution specified by the network can now be calculated as follows:

$$P_\phi(y) = \frac{1}{Z} \prod_{i=1}^{N} \phi_i(y_i) \prod_{(ij)\in E} \phi(y_i, y_j), \tag{7}$$

where $Z = \sum_{y'} \prod_{i=1}^{N} \phi_i(y'_i) \prod_{ij\in E} \phi_{ij}(y'_i, y'_j)$ is the partition function. The basic issue, the maximum a-posteriori (MAP) inference problem in a Markov network is to find:

$$arg\ max_y P_\phi(y). \tag{8}$$

In general, the node and edge potentials are functions of the features of the objects $x_i \in \Re^{d_n}$ and

features of the relationships between them $x_{ij} \in \Re^{d_e}$. We define the number of node features $d_n$ and the number of edge features $d_e$.
In order to formulate the dependence of the potentials on the features in a way, that is likely to be simple, a log-linear combination of the feature-vector with a label-specific row vector is applied:

$$log\phi_i(y_i) = \sum_{k=1}^{K}(w_n^k \cdot x_i)y_i^k, \text{ for nodes and}$$
$$log\phi_{ij}(y_i, y_j) = \sum_{k,l=1}^{K}(w_e^{k,l} \cdot x_{ij})y_i^k y_j^l, \text{ for edges,} \quad (9)$$

$w_n^k$ and $w_e^{k,l}$ are the label-specific row vectors of size $d_n$ and $d_e$, that are assigning weights to the feature vectors. Coming back to $y$, described above as possible assignment of values to $Y$. $y$ is now represented as a set of $K \cdot N$ indicators $y_i^k$, where $y_i^k = I(y_i = k)$. This leads to the following equation:

$$logP_w(y|x) = \sum_{i=1}^{N}\sum_{k=1}^{K}(w_n^k \cdot x_i)y_i^k + \sum_{(ij)\in E}\sum_{k,l=1}^{K}(w_e^{k,l} \cdot x_{ij})y_i^k y_j^l - logZ_w(x), \quad (10)$$

defining the log of the conditional probability of $P_w$. The partition function $Z$ only depends on $w$ and $x$, but not on the labels $y$.
Replaced by a more compact notation, equation (10) can be rewritten as:

$$logP_w(y|x) = wXy - logZ_w(x), \quad (11)$$

where $w = (w_n, w_e)$ and $y = (y_n, y_e)$. The matrix $X$ consists of the node feature vectors $x_i$, the edge feature vectors $x_{ij}$, both repeated multiple times. At least it is appropriatly padded with zeros.
Finally, the MAP assignement is now given by:

$$arg\ max_y logP_w(y|x), \quad (12)$$

which simply corresponds to maximizing the linear function $wXy$.

## 4.2 Markov Chain Monte Carlo Sampling

The concept of *sampling* stands for: “randomly select one element out of many”. The sets we have to treat with are often way too big for efficiently applying trivial sampling methods that are producing all entries and then randomly select one element. Therefore, using Markov chains for sampling increases the applicability. A sequence $(X_1, ..., X_n)$ of random variables $X_i$, taking values in $X$ is called a Markov chain, if it fulfills the Markov property given as:

$$P(X_n = x_n|X_1 = x_1, ..., X_{n-1} = x_{n-1}) = P(X_n = x_n|X_{n-1} = x_{n-1}), \forall n \geq 1 \text{ and } x_1, ..., x_n \in X. \quad (13)$$

This phenomenon is also called the *memoryless property*, because the conditional distribution of $X_n$ depends only on $X_{n-1}$. After all, the random selection of elements from one sampling step to another can also be improved if the chance for “better” elements becomes more and more probable.

An overview of Finite Random Fields is presented to elementary introduce the basic ideas behind sampling from a given target distribution.
Gibbs-Sampling is a special class of MCMC algorithms, which creates a sequence of samples from the probability distributions of two or more random variables in order to approximate the unknown joint distribution.

### 4.2.1 Finite Random Fields

A finite index set $S = \{1, 2, ..., N\} \times \{1, 2, ..., N\}$ is the set of $N^2$ pixels, called sites in the 2D case. For every site $s \in S$ there is a finite space $X_s$ of states $x_s$. The neighborhoud $\mathbb{N}(s)$ for a fixed site $s$ is defined as the number of 4-adjacent pixels to $s$. The space of configurations $x = (x_s)_{s \in S}$ is defined: $X = \prod_{s \in S} X_s$. Probability measures or distributions $P$ on $X$ can be represented as $P = (P(x))_{x \in X}$ such that $P(x) \geq 0$ and $\sum_{x \in X} P(x) = 1$. Having a subset $E \subset X$, the probability of $E$ is given by: $P(E) = \sum_{x \in E} P(x)$.
If $P$ is a positive probability measure on $X$, then $P$ is a *random field*.
The random field $P$ is a *Markov field* if for all $x \in X$,

$$P(X(s) = x_s | X(S \setminus s) = x_{S \setminus s}) = P(X(s) = x_s | X(\mathbb{N}(s)) = x_{\mathbb{N}(s)}). \tag{14}$$

A set of random variables $X(S)$ is said to ba a Gibbs random field (GRF) on $S$, if and only if its configuration obey a Gibbs distribution. This is defined as:

$$P(X) = \frac{1}{Z} exp\left\{-\sum_{c \in C} \phi_c(x_c)\right\}, \text{ with } Z = \sum_x exp\left\{-\sum_{c \in C} \phi_c(x_c)\right\}. \tag{15}$$

One theorem that describes the equivalence of these two types of properties is the so-called *Hammersley-Clifford theorem*. It states that a random field is a MRF on $S$ with respect to $\mathbb{N}$, if and only if it is a GRF on $S$ with respect to $N$.

### 4.2.2 Ising Model

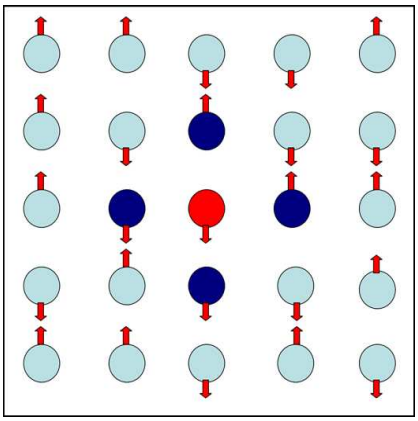


Figure 15: Ising model with two types of sites: up (+1) and down (-1).

The Ising model has been proposed by Ernst Ising in his doctoral thesis. It is a model for explaining magnetic behaviour, considering an idealised system of interacting particles. Figure 15 shows such a system on a regular planar grid. Each particle $x_i$ interacts only with its four nearest neighbors (see Figure 14). To compute the total energy of a system, all spin orientations are needed. The probability of any particular configuration $c$ is:

$$\frac{1}{Z} exp\left\{-\frac{E(c)}{kT}\right\}, \text{with } E = -J \sum_{x_i \sim xj} x_i x_j \text{ and } Z = \sum_{c \in C} exp\left\{-\frac{E(c)}{kT}\right\}, \tag{16}$$

where $x_i \sim x_j$ are adjacent particles, $C$ is the set of possible configurations, $k$ is Boltzmann‘s constant, T is the absolute temperature in K and $E(c)$ is the energy of configuration $c \in C$. The constant $J$, when positive, decreases the total energy for an agreement between $x_i$ and $x_j$. At low temperatures, the most likely states are those with lowest energy.

In our case, we consider binary images $I$, having two kinds of pixel-coverage, black and white. It is an obvious suggestion that the probability for pixel $x_i$ of beeing white increases with more white adjacent pixels then black ones.

$$P(x_i = w|x_{-i}) = \frac{exp(\beta n_i^w)}{exp(\beta n_i^w) + exp(\beta n_i^b)}, \tag{17}$$

where $x_{-i} = \{x_j : j \in I \setminus i\}$, $n_i^w = \sum_{j\sim i} I_{x_j=w}$ the number of white adjacent pixels of $x_i$ and:

$$\phi_c(x_c) = \begin{cases} -\beta, & \text{both pixel in clique C have the same color} \\ 0, & \text{otherwise} \end{cases},$$

Considering equation (18), this leads to:

$$P(x) = exp\left\{\beta \sum_{i\sim j} I(x_i)I(x_j)\right\} / Z(\beta), \text{ with: } Z(\beta) = \sum_x exp\left\{\beta \sum_{i\sim j} I(x_i)I(x_j)\right\}. \tag{18}$$

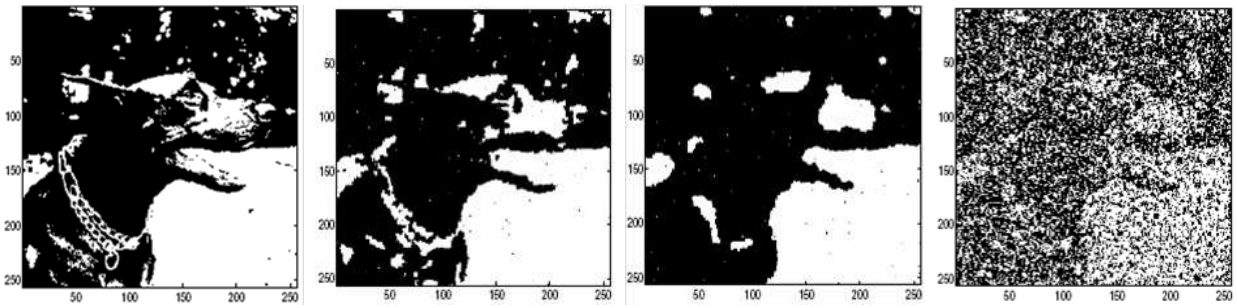
### 4.2.3 Gibbs Sampling

Sampling from a Gibbs field $P(x) = \frac{1}{Z}exp(-E(x))$ with $E(x) = -\beta \sum_{i\sim j} I(x_i)I(x_j)$ for the Ising- and $E(x) = \beta \sum_{i\sim j}(I(x_i) - I(x_j))^2$ for the Potts model (see Chapter 4.2.5), is the basis of minimum mean squares estimation, since the estimate is the mean of samples. Table 2 shows the pseudo-code notation of the Gibbs sampler for the Ising model:

```
Gibbs-Sampler for the Ising model
choose x = w, x = b or all pixels randomly;
for t = 1 : n_iter
    for j = 1 : n
        i = adjacent pixel to j;
        p = exp(βn_i^w)/exp(βn_i^w) + exp(βn_i^b);
        if rand(1) < p
            x_i = b;
        else
            x_i = w;
        end
    end
end
```

Table 2: Gibbs sampler for the Ising model; $n_{iter}$ denotes the number of iterations and $n$ the absolute number of pixels.

Figure 16 shows an example image for the Ising model.



(a) (b) (c) (d)

Figure 16: (a) original image; (b) after 500,000 (c) 5,000,000 steps and (d) after 300,000. Panels (b) and (c) have $\beta = 0.85$, while panel (d) has $\beta = 0.25$; [11].

### 4.2.4 Metropolis Algorithm

Another common sampling method is the Metropolis algorithm. This would have the following steps:

| Metropolis-Sampler for the Ising model |
|---|
| for $t = 1 : n_{iter}$ |
| for $j = 1 : n$ |
| $i$ = adjacent pixel to $j$; |
| $x_i' = 1 - x_i$; |
| $nagree = \beta \sum_{j \sim i} I(x_i == x_j)$; |
| $ndisagree = \beta \sum_{j \sim i} I(x_i' = x_j)$; |
| $p = exp(min\{0, ndisagree - nagree\})$; |
| if $rand(1) < p$ |
| $x_i = x_i'$; |
| end |
| end |
| end |

Table 3: Metropolis sampler for the Ising model; $n_{iter}$ denotes the number of iterations and $n$ the absolute number of pixels.

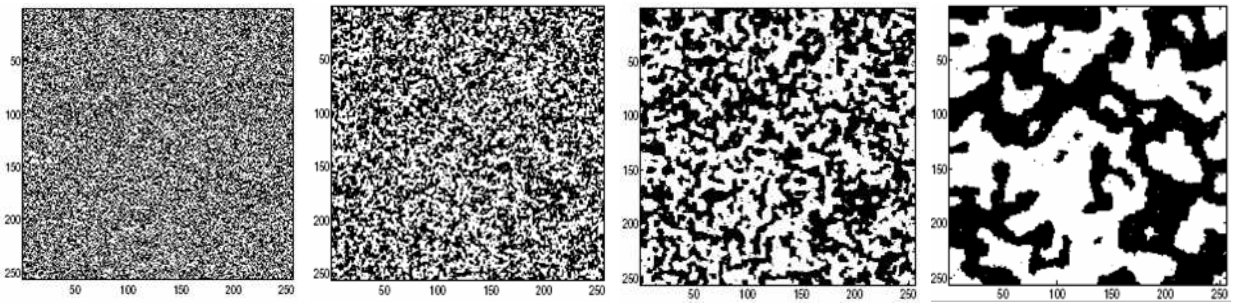
Figure 17 shows an example for the Ising model via Metropolis algorithm.

### 4.2.5 Potts Model

The Potts model is a generalization of the Ising model for more than two colors with $x_i \in \{0, 1, 2, ..., n_c - 1\}$. The energy function for the Potts model is given as:

$$E(x) = \beta \sum_{i \sim j} (I(x_i) - I(x_j))^2, \text{ with: } \beta > 0. \tag{19}$$

If $I(x_i) = \pm 1$ then $-I(x_i)I(x_j) = (I(x_i) - I(x_j))^2/2 - 1$ and therefore equation 19 is compatibel with the binary Ising model (see equation 18).



(a) (b) (c) (d)

Figure 17: (a) Random Start; (b) after 100,000 (c) 500,000 (d) 5,000,000 steps of Metropolis algorithm with inverse temperature $\beta = 0.85$; [11].

## 4.3 Associative Markov Networks

Associative Markov networks(AMNs) are a subclass of pairwise Markov networks that have an important refinement. They do model those problems, where related variables tend to have the same label. Furthermore, AMNs allow different labels to have different attraction strengths. This is represented in the following constraints: $w_e^{k,l} = 0$ for $k \neq l$ and $w_e^{k,k} \geq 0$, which results in $\phi(k,l) = 1$ for $k \neq l$ and $\phi(k,k) = \lambda_{ij}^k$, where $\lambda_{ij}^k \geq 1$. This makes AMNs a generalization of the previously explained Potts model. The basic idea behind these conditions is, that edges between nodes with different labels should be penalized over edges between equally labeled nodes.
The linear programming relaxation of the MAP problem for these networks can be written as:

$$max \sum_{i=1}^{N} \sum_{k=1}^{K} (w_n^k \cdot x_i) y_i^k + \sum_{(ij) \in E} \sum_{k=1}^{K} (w_e^{k,k} \cdot x_{ij}) y_{ij}^k \tag{20}$$

$$\text{so that:} \quad y_i^k \geq 0, \quad \forall i,k; \quad \sum_k y_i^k = 1, \quad \forall i;$$

$$y_{ij}^k \leq y_i^k, \forall i,k; \quad y_{ij}^k \leq y_j^k, \quad \forall (ij) \in E, k.$$

## 4.4 Learning with Associative Markov Networks

The starting-point for the learning phase is having one single feature vector $x$. It includes all $x_i$, where each $x_i$ consists of non-zero values only, representing all features to the belonging datapoint $p_i$. The MAP problem formulates this task as finding the set of labels $y$ that maximizes the log of the conditional probability $logP_w(y|x)$.
The main task for the learning phase is to find the weights $w^*$, such that:

$$w^* = arg\ max_w logP_w(\hat{y}|x), \tag{21}$$

where $\hat{y}$ defines the vector of correct labels. The inference is then denoted by:

$$y^* = arg\ max_y logP_{w^*}(y|x). \tag{22}$$

An alternative method introduced in [2] is to maximize the margin of confidence in the true label assignment $\hat{y}$ over any other assignment $y \neq \hat{y}$, represented by:

$$logP_w(\hat{y}|x) - logP_w(y|x) = wX(\hat{y} - y). \tag{23}$$

The advantage of this maximum margin optimization is, that the term $Z_w$ cancels out and the maximization can be done efficiently.

# 5 Feature Extraction

This chapter presents the different types of feature extraction methods that have been implemented for this work. In Figure 2, which gives an overview on the structure of this work, the inner part of the second bounding-box is regarded, inlcuding all feature extraction methods that are embedded into the AMN (see Chapter 4).
Problems like splitting touching objects of the same class have to be solved. Therefore, a new approach of curvature estimation is explained.

## 5.1 Smoothing Algorithm

As described in Chapter 2, connected component labeling is applied to the data that has been calculated using voxel-wise gray-scale invariants. Figure number 18(left) shows an example. In order to close improper holes inside the objects or to better approximate the objects‘ appearance towards cell shapes, a smoothing algorithm is applied.
The actual implementation of the algorithm follows the theory of the Ising model, introduced in the previous chapter. Figure 18(right) and 19(right) show an example result of the 2D algorithm, applied to one slice and of the 3D variant, applied to the whole datavolume.

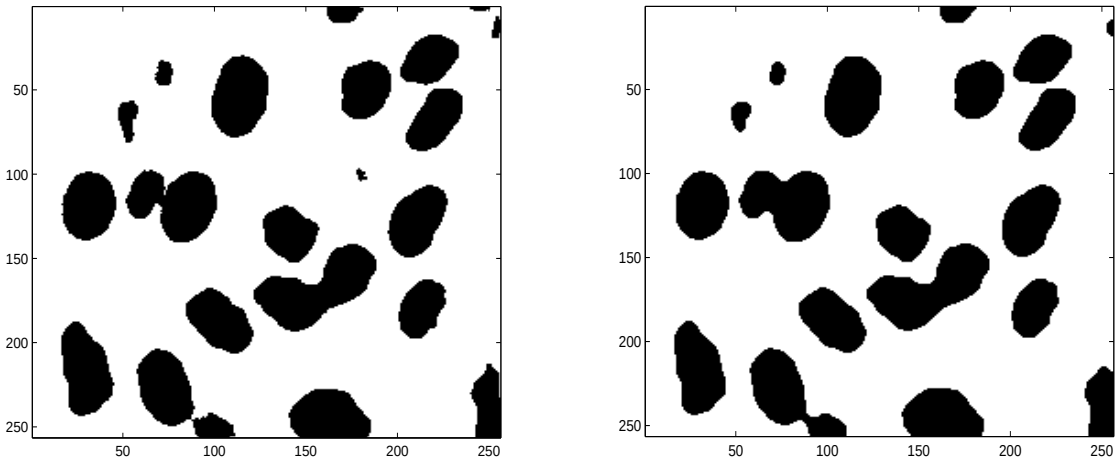



Figure 18: Smoothing algorithm in 2D; Left: Original data; Right: Smoothed data.

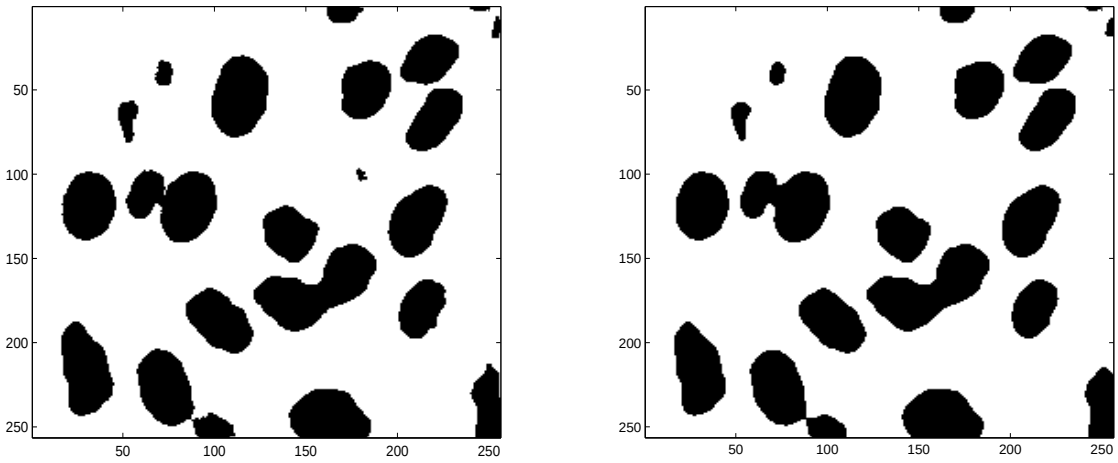



Figure 19: Smoothing algorithm in 3D; Left: Slice of the original data-volume; Right: Slice of the smoothed data.

## 5.2 Toy Data

In order to test the implemented algorithms and to show the results of each calculation step more evidently, a representative set of toy examples is needed for 2D and 3D data. One requirement for a toy example is a proper representation of a typical cell-couple that needs to be divorced during the calculations. This is realized by the example shown in Figure 20.

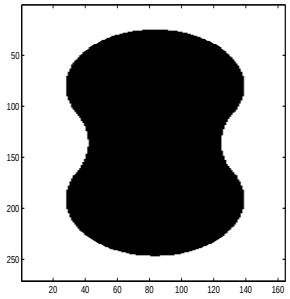


Figure 20: Toy-Example.

To achieve better knowledge about the effects of the different features, some more datasets are used in 2D and 3D. These sets are from the Kimia silhouette database[2] and the Princeton Shape Benchmark[3].

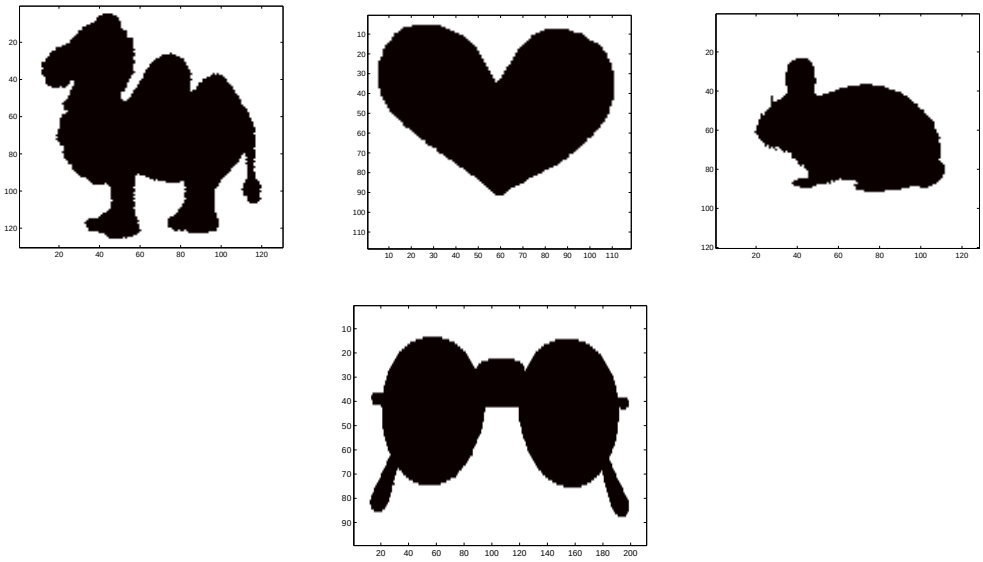


Figure 21: 2D-Toy-Dataset.

Figure 22: 3D-Toy-Dataset.

[2] http:://www.lems.brown.edu/vision/software/index.html
[3] http://shape.cs.princeton.edu/benchmark/

## 5.3 Fast Local Curvature Estimation

Local curvature (of a surface) is a valuable rotational invariant feature for many computer vision and pattern recognition problems in 3D. For example, it has been shown in [12], that it is possible to reconstruct (up to a constant factor) an object surface from the principle curvatures of each surface point. This makes curvature a useful shape descriptor which can be applied i.e. for object recognition or segmentation tasks.

For a continuous setting in $\mathbb{R}^2$, where the object surface is represented by a planar curve $C$, and given a suitable parameterisation of $C$, $k$ can easily be calculated via derivation. So curvature appears to be a measure, that is very easy to handle. However, taking a look at surfaces in $\mathbb{R}^3$ shows that there is no canonical definition of curvature at all. Several different measures have been introduced in literature, most of them relying on the so called *principal curvatures* $k_{max}$ and $k_{min}$. Following the definition in $\mathbb{R}^2$, $k_{max}$ and $k_{min}$ are computed as the extreme values of the 2D curvatures which are embedded in all possible planes spanned by the normal vector and the tangents in the evaluation point of the surface. The most common measures are the *mean curvature*, given as $H := (k_{max} + k_{min})/2$ and the *Gaussian curvature*, $K := k_{max} \cdot k_{min}$.
Given the previous definitions, curvature is easy to handle in continuous settings, but for discrete gray-scale images and volume data, things turn out to be a lot more difficult.

First of all, an overview on curvature measures is given. Starting with a planar continuous setting, for a given contour parameterized as $S(t) = (x(t), y(t))$, curvature is essentially defined via the second order derivatives with respect to $t$:

$$k := \left| \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}} \right| \tag{24}$$

This general differentiation approach is well suited for any continuous setting, but as we stressed before, in the discrete case this usually suffers from the fact that contour reconstructions from discrete data is always nothing more than an approximation of the original contour, which additionally suffers under noise.
In order to overcome these problems, we consider a second, rather intuitive definition of curvature: for a plane curve $S$, the curvature at a given point $p \in S$ is defined as the reciprocal of the radius of an osculating circle (see Figure 23).

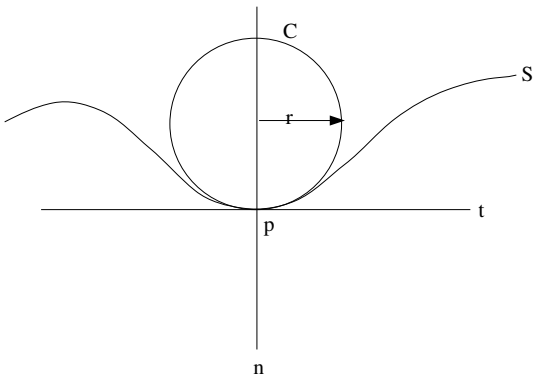



Figure 23: For a continuous setting in $\mathbb{R}^2$, local curvature $k$ at a given point $p$ is defined as $k = 1/r$.

This intuition is backed by some rather simple facts: considering a circle $C$ with radius $r$ parameterized in arc length. $C$ is given as $C = re^{it/r}$. Applying the differential approach, we see that the curvature is just $1/r$, which is equal to the absolute value of the 2nd derivative. This relation, along with the properties of (24) shows, that the second derivative is the dominant factor in local curvature estimation. Now, for a given point $p$ on some contour $S$, which is also parameterized in arc length, we consider osculating circles at $p$. With full osculation at $p$, we can neglect the first derivatives when expanding the Taylor series:

$$C(t) = 1/rt^2 + O(t^3) \tag{25}$$
$$S(t) = S''(0)t^2 + O(t^3) \tag{26}$$

Since the curvature for circles is given, we now estimate the osculation in order to find the circle, which delivers the best local curvature approximation and choose it's curvature as local approximation at $p$:

$$\|S(t) - C(t)\| \leq \|(S''(0) - 1/r)\|t^2 + O(t^3) \tag{27}$$

Starting from this relation, we can estimate the local curvature at $p$ by the radius $r$ which minimizes (27), neglecting higher order derivatives. The expected error of this approximation is bounded in terms of $O(t^3)$.
Following our approach, we have to find the appropriate circle $C$ which minimizes (27). By definition, $C$ has it's center on the normal $n$ at $p$ (see fig. 23) reducing the optimization problem to a simple convex search over $r$. All we need is an appropriate osculation measure.
For our discrete problems, we can find such a measure which is very robust and easy to implement: we simply integrate over the point-wise distance of the circle surface with the object contour (see fig. 24).

$$\int_t \delta(\|C(t) - S(t)\|)dt \tag{28}$$

If we choose $C$ not to be binary, but real valued and normalized to one, we can directly compare circles of different radii and choose the argmax as best fit.

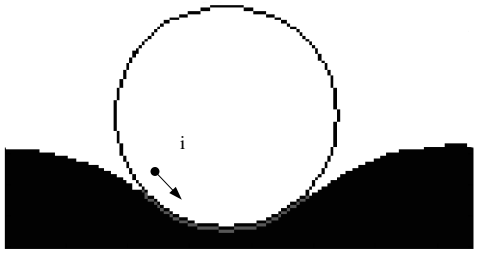



Figure 24: Schema of the integration over the point-wise distance of the circle surface with the object contour.

### 5.3.1 Fast Integral Curvature Measure

For the case of a binary 2D image $I$, discrete images of circles $C_r$ (see Figure 25) with radius $r$ are computed previously and normalized to one ($\sum C_r = 1$). Then the algorithm follows a multi scale approach, which is applied using different radii separately.

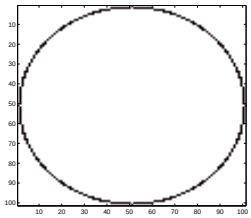


Figure 25: Circle image $C_{50}$.

First, the input image $I$ is convolved with $C_r$. This convolution is equal to a point-wise integration over a spherical neighborhood of the respective radius.

$$I_{Ir} = I * C_r$$

So the resulting image $I_{Ir}$ stores the integration results for all possible center points:

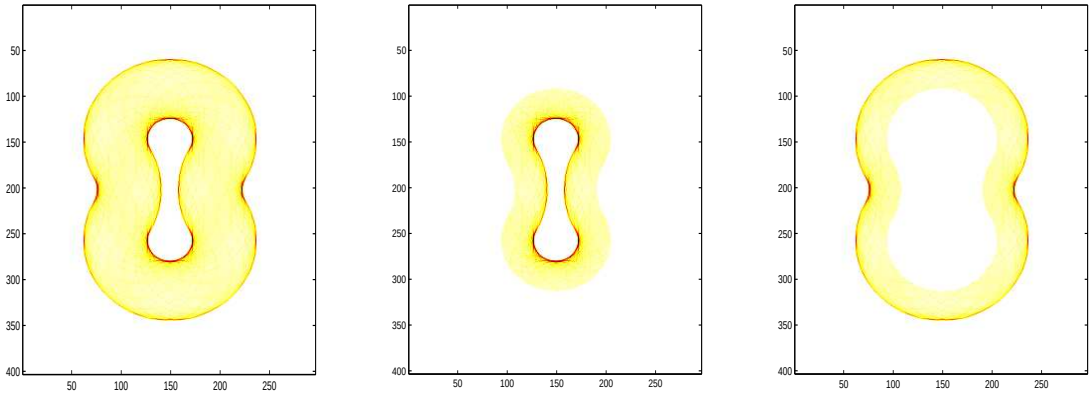


Figure 26: Convolution result of the toy-example. Left: $I_{Ir}$; Center: $I_{Ir+}$; Right: $I_{Ir-}$.

In the next steps, a copy $I'_{Ir}$ of $I_{Ir}$ is made, where those entries over a certain threshold $t$ are set to one and all others are set to zero (see Figure 27).

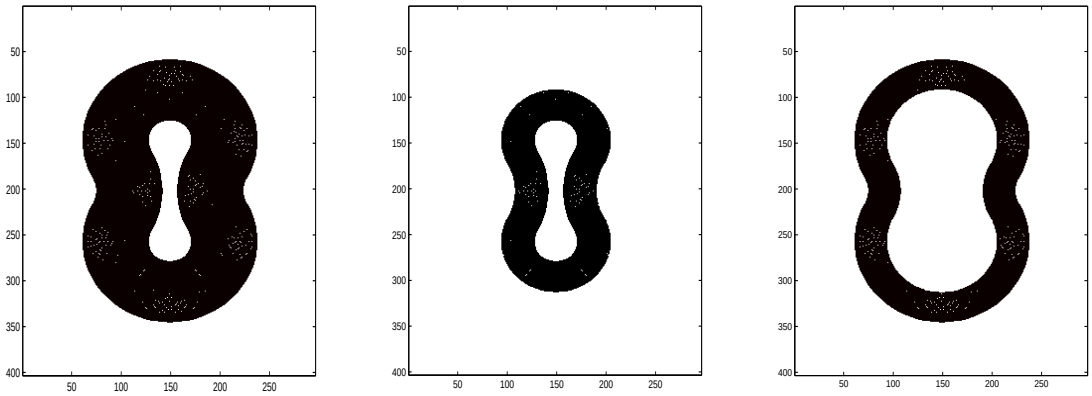


Figure 27: Normalizing mask for the convolution result of the toy-example. Left: $I'_{Ir}$; Center: $I'_{Ir+}$; Right: $I'_{Ir-}$.

Then $I_{Ir}$ is split in a convex ($I_{Ir+}$) and concave ($I_{Ir-}$) part, depending on whether the center point lies within the original object or not. $I_{Ir-}$ is weighted with $-1$ to denote concavity. Afterwards, the integration results are back-projected onto possible points of the contour which again can be achieved via fast convolution:

$$I_{Pr+} = (I_{Ir+} * C_r) \cdot I_C$$

$$I_{Pr-} = (I_{Ir-} * C_r) \cdot I_C$$

where $*$ denotes a convolution and $\cdot$ a point-wise multiplication. Figure 28 shows both back-projection results.

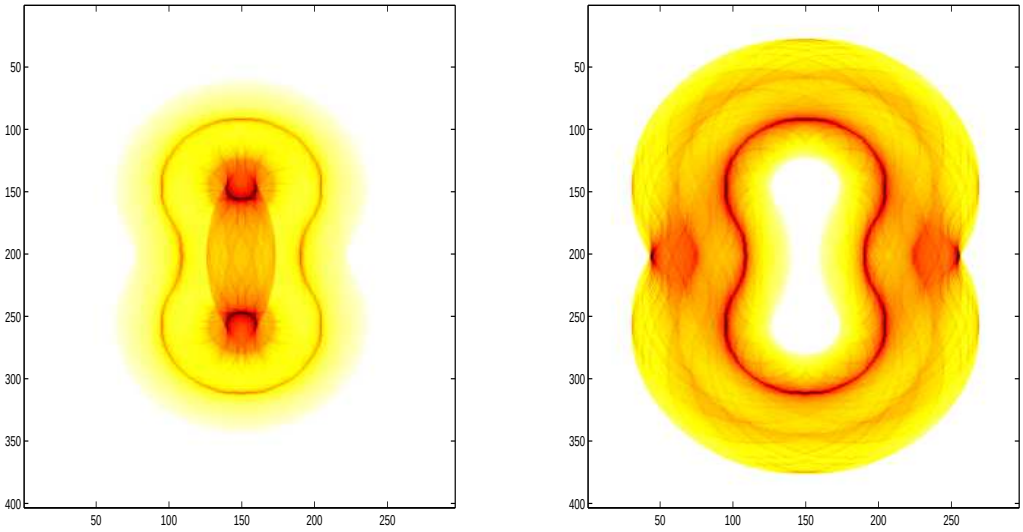



Figure 28: Back-projection result of the toy-example.

At this point, it becomes more clear, why $I'_{Ir}$ has been calculated before. In Figure 29, $I_{Pr+}$ and $I_{Pr-}$ is masked with the contour of the input image $C_I$. Here, it can easily be seen that without further normalization the results would not be satisfying.

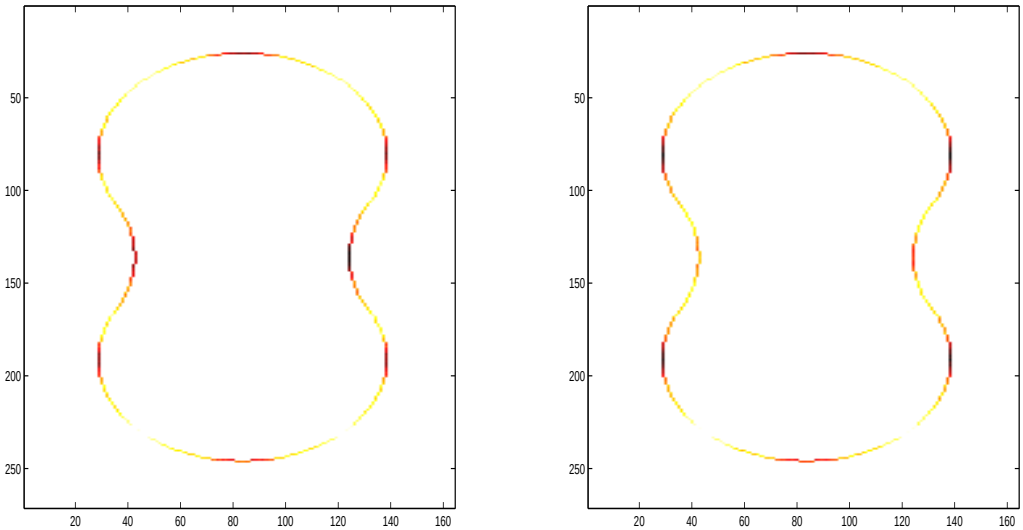



Figure 29: Result of the toy-example without normalization.

Therefore, the normalized convolution-results are also back-projected:

$$I'_{Pr+} = (I'_{Ir+} * C_r) \cdot I_C$$

$$I'_{Pr-} = (I'_{Ir-} * C_r) \cdot I_C$$

This results in the following Figure:

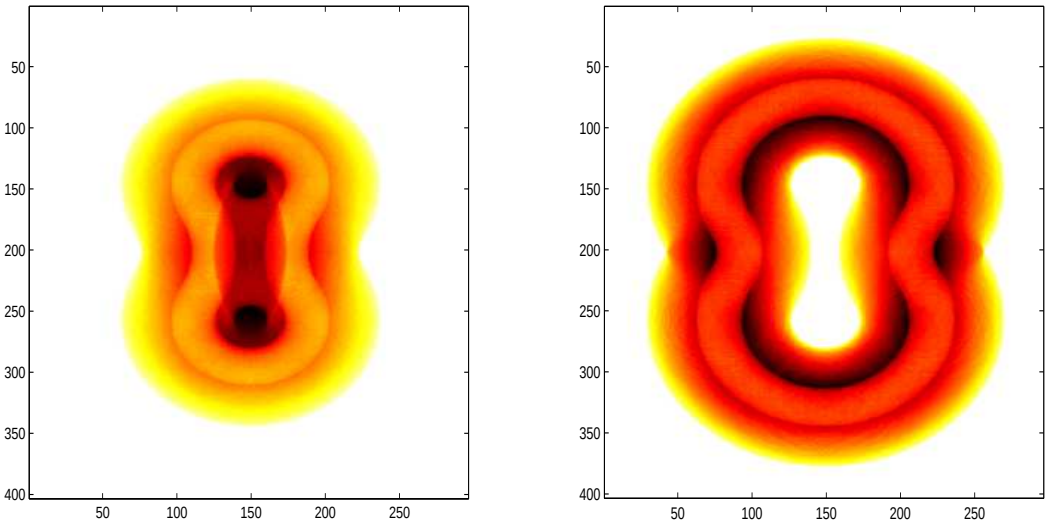



Figure 30: Normalizing mask for the back-projection result of the toy-example. Left: $I'_{Pr+}$; Right: $I'_{Pr-}$.

After normalizing $I_{Pr+}$ with $I'_{Pr+}$ and $I_{Pr-}$ with $I'_{Pr-}$ the following results are achieved:

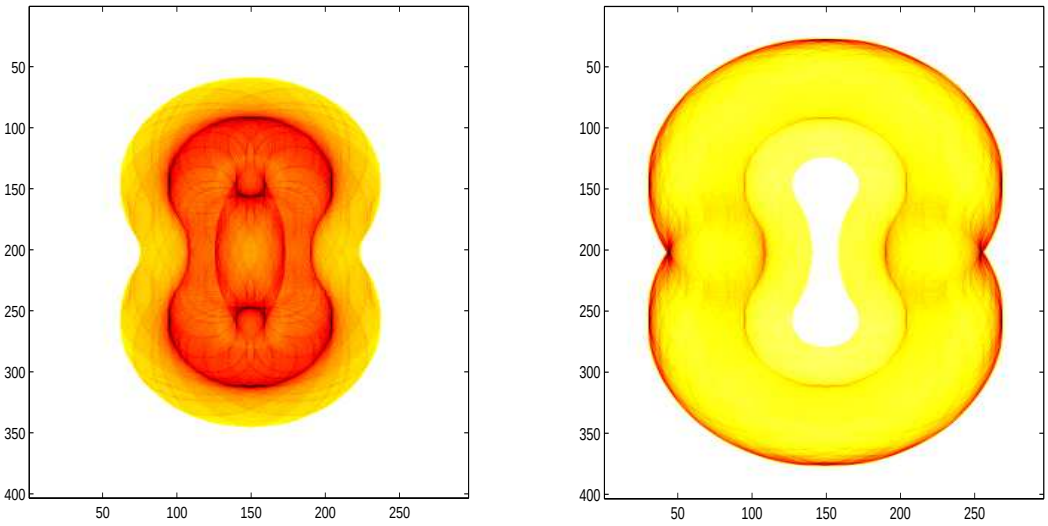



Figure 31: Normalized back-projection result of the toy-example.

In a final step, the convex and concave parts are reassembled and restricted to the original contour.

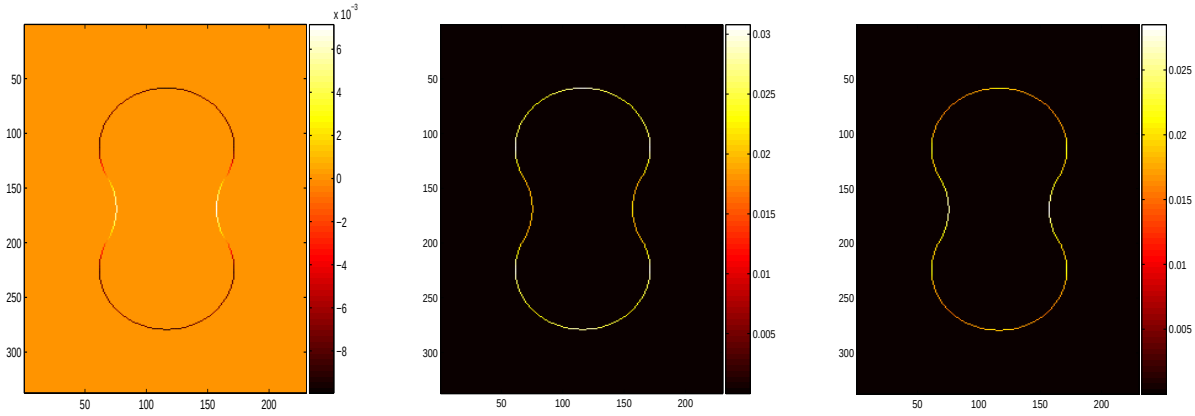


Figure 32: Normalized result of the toy-example with original scaling.

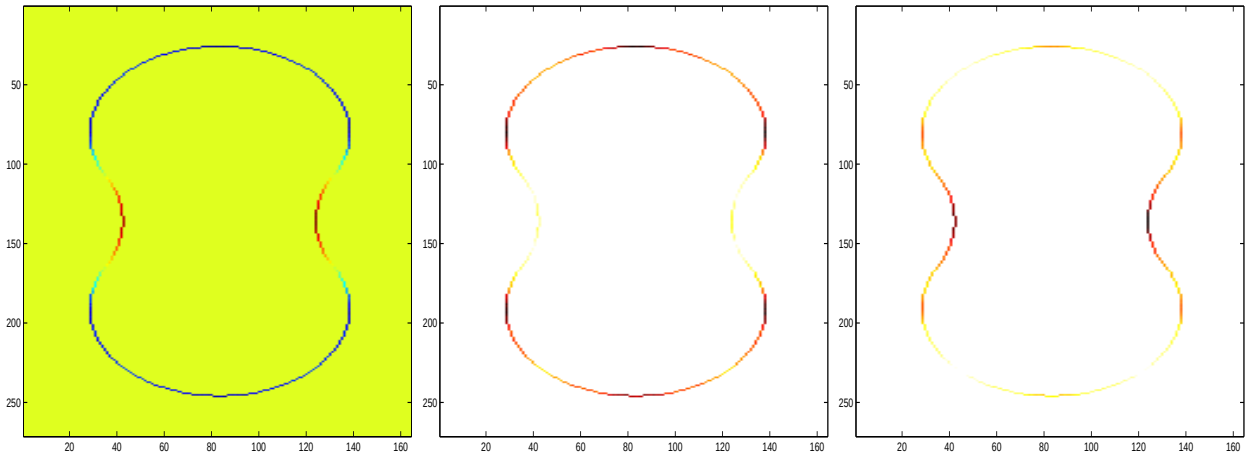


Figure 33: Normalized result of the toy-example with different scaling.

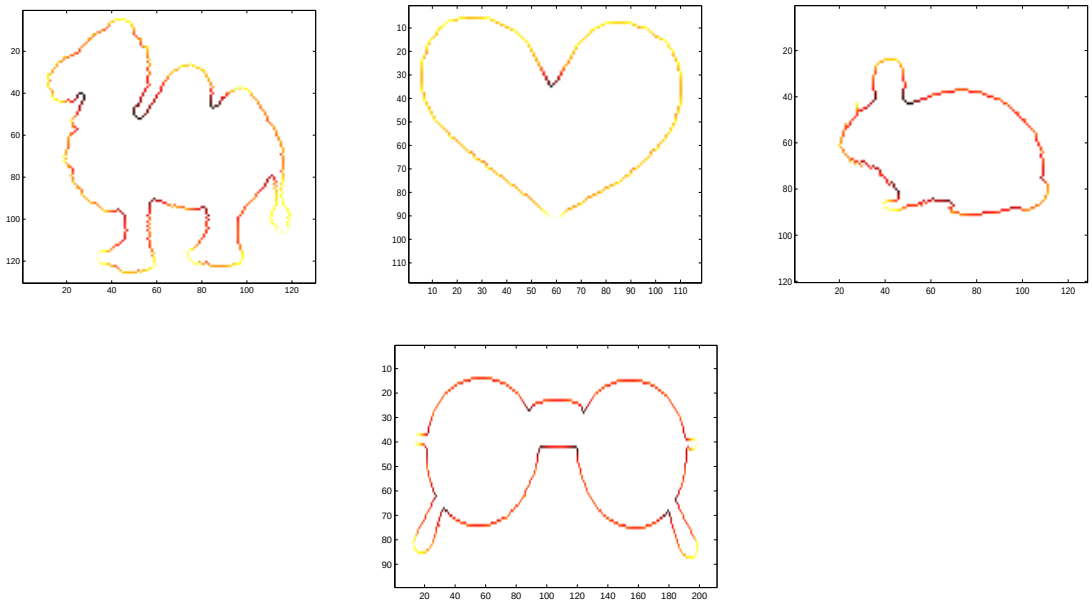


Figure 34: Toy-dataset results.

Applying this curvature estimation for multiple scales, and choosing the curvature given by the optimal radius in terms of the maximal back-projection value at all contour points, leads to very reliable curvature estimates. Some representative results of the 2D binary toy dataset are shown in Figure 34.

### 5.3.2 Noise Robustness

A key property for curvature estimation features is it's robustness towards noise. The following two Figures show the results of this approache, applied to the toy-example with 50% and 95% of salt- & pepper noise.

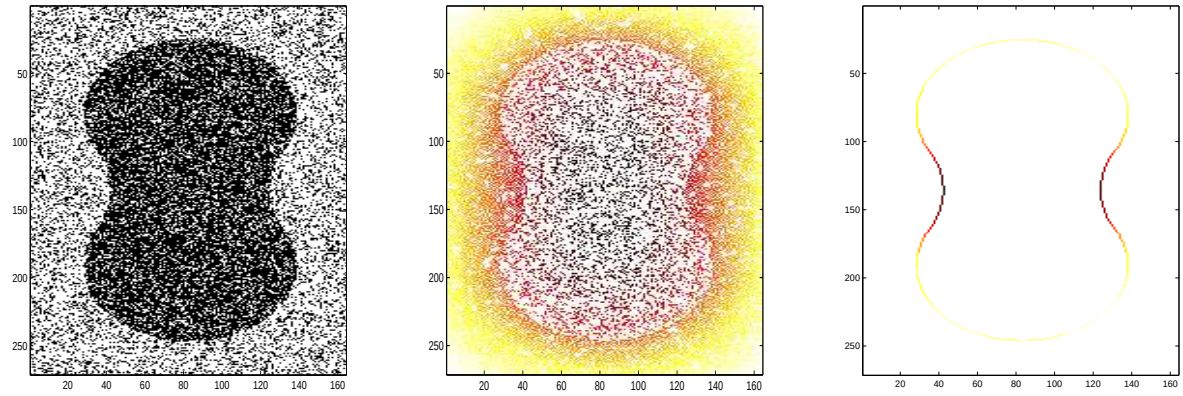


Figure 35: Result of the toy-example with 50% salt & pepper noise.

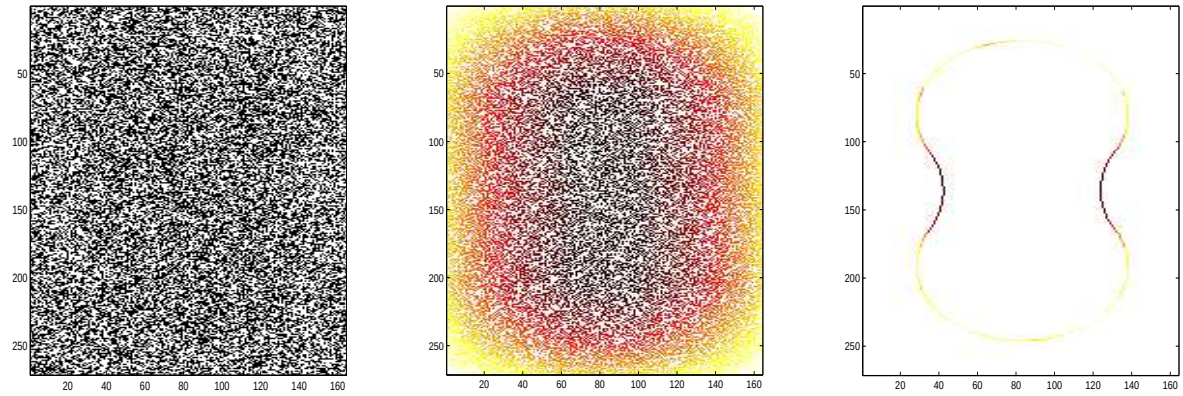


Figure 36: Result of the toy-example with 95% salt & pepper noise.

From a theoretic point of view, it is evident that integration should be far more robust towards noise than differentiation. Figure 37 illustrates the behaviour of this approach under noise. It can be seen, that the results are quite tractable even if 90% of salt- & pepper noise is applied to the toy-example.

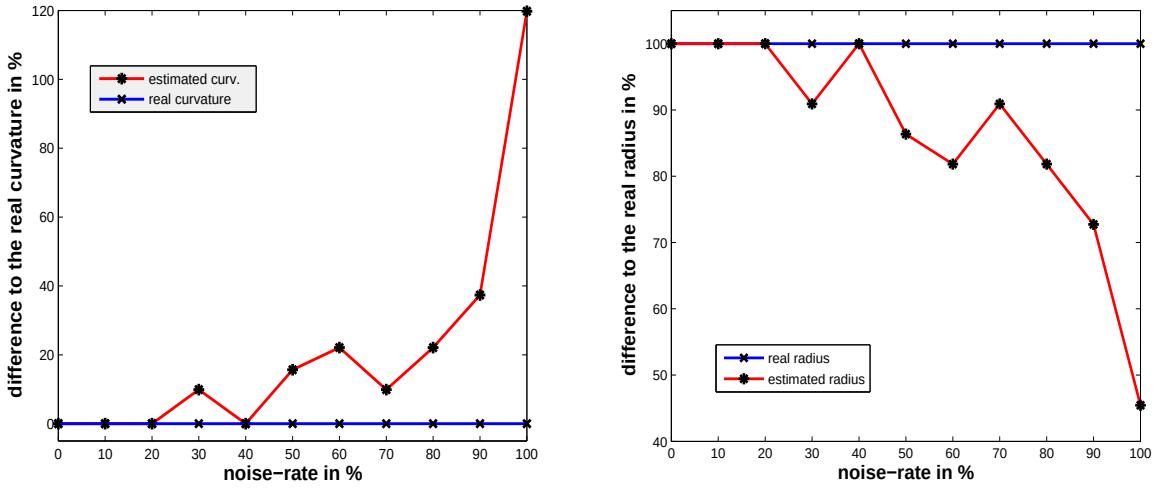



Figure 37: Result of the toy-example with salt & pepper noise from 0 to 100%.

### 5.3.3 3D binary volume data

The extension from the binary case in 2D to 3D is straight forward: instead of circles, we simply apply spheres in the convolution algorithm. Then Gaussian or mean curvature is computed by weighting the curvature estimate by the integration results of the different spheres.

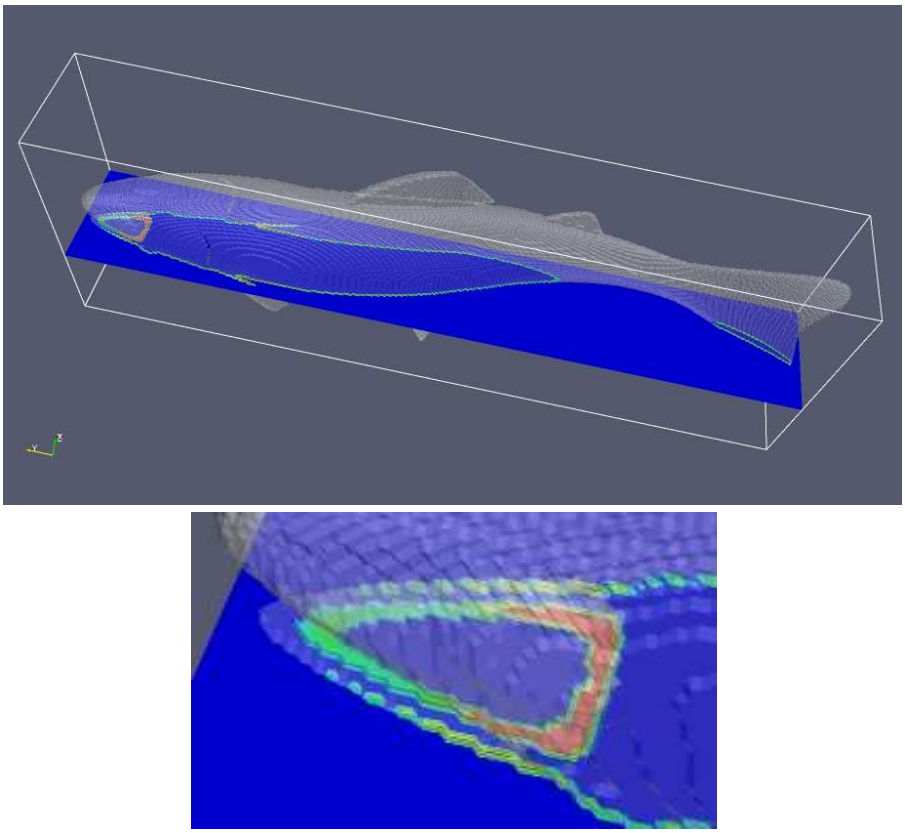


Figure 38: Result of the 3D toy-dataset.

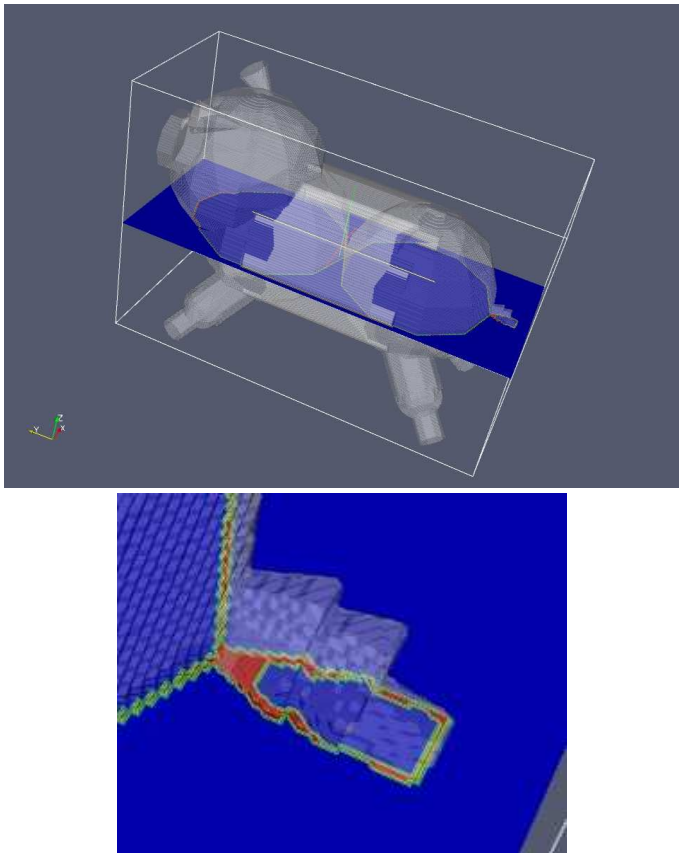


Figure 39: Result of the 3D toy-dataset.

### 5.3.4 Cell Database

At last, the local curvature estimation method is applied to the cell database introduced in Chapter 2. Figure 40 shows an example on the left and the calculated curvature result on the right.

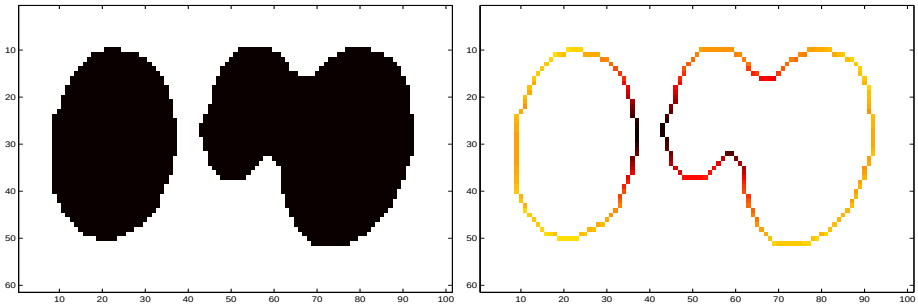



Figure 40: Results of the local curvature estimation on a cell dataset.

At this point, another problem appears. It is true, that the significant areas, where two cells overlap, are detected. But furthermore, those regions, where two singular objects are lying close to each other are also marked as having high curvatures. This problem arises exactly at those areas, where the distance between the objects is nearly equal to the chosen radius of the convolution sphere (circle) $C_r$. Even with a multi-scale approach, using different radii, this problem could not be handled satisfyingly.
Taking a look at Figure 40, a possible solution arises. Those areas, where the curvature estimation failed are concave object regions, contrary to those areas of overlapping cell-objects.

## 5.4 Convexity Calculations

It is easy to achieve the measure of convexity for the case of 2D binary data. For every pixel $p_1$ on the contour of an object, convexity (concavity) is calculated by detecting two corresponding pixels $p_2, p_3$ that lie in a previously determined distance in opposite directions on the contour. A 3x3 matrix $M$ is formed from the three pixel coordinate vectors, by appending the unity vector.

$$M = \begin{pmatrix} x_2 & y_2 & 1 \\ x_1 & y_1 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}$$

The determinant of this matrix is, when positive, an appropriate measure for the degree of convexity, or, when negative, the concavity of the contour.

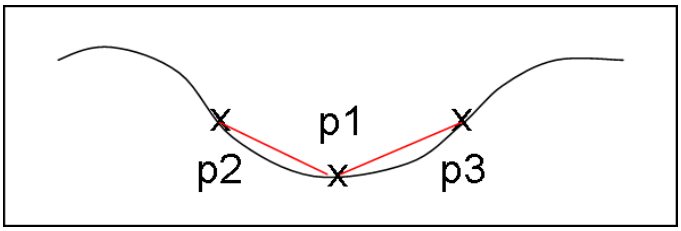



Figure 41: Convexity calculation.

Figure 42 shows some results of the convexity calculations applied to the 2D toy-dataset.

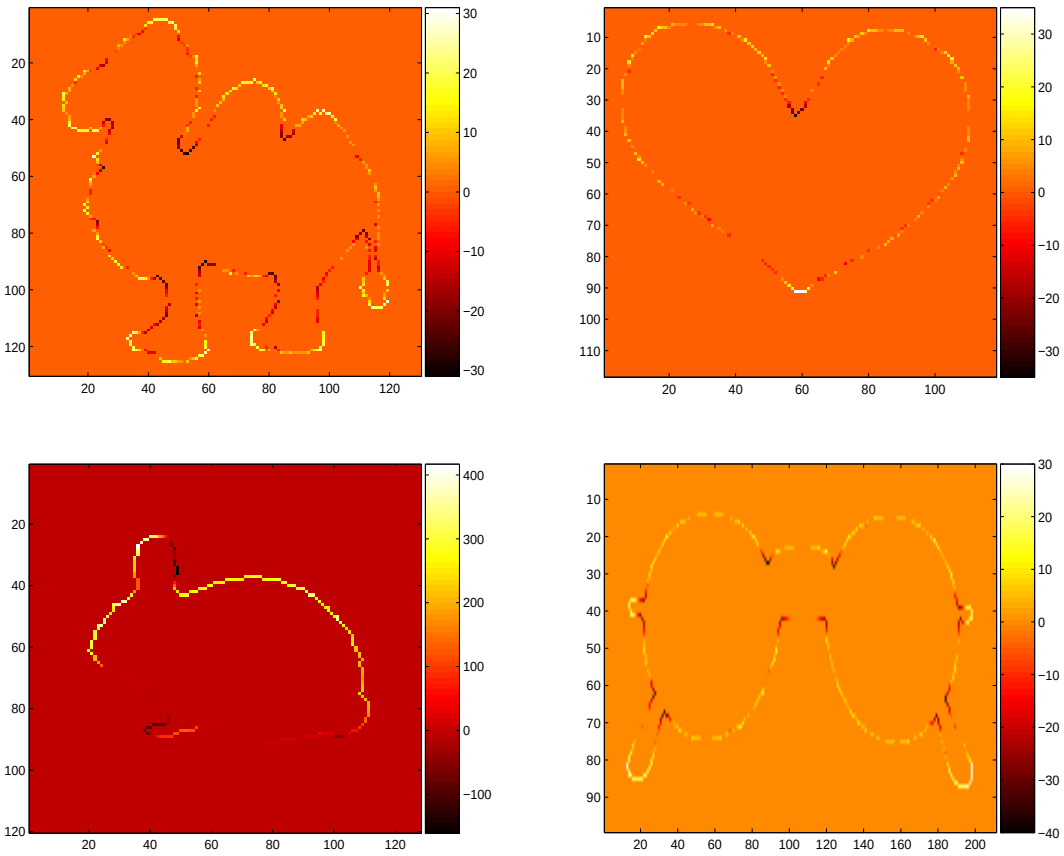


Figure 42: Toy-dataset results.

## 5.5 Splitting Algorithm

Finally, an algorithm is needed, that splits an object at a marked position. The first step is calculating the distance transform of the binary object $P$.

$$\langle P\rangle = \{p : p \in \mathbb{G} \wedge P(p) = 1\} = P^{-1}(1), \text{ and}$$
$$\langle \overline{P}\rangle = \{p : p \in \mathbb{G} \wedge P(p) = 0\} = P^{-1}(0),$$

are proper subsets of $\mathbb{G}$ where $\mathbb{G}$ is the grid on which $P$ is defined. The $d_\alpha$ distance transform $T$ of $P$ associates with every pixel $p$ of $\langle P\rangle$ the $d_\alpha$ distance from p to $\langle \overline{P}\rangle$. Here, $d_\alpha$ is the 4-adjacent grid point metric, defined as:

$$d_4(p,q) = |x_1 - x_2| + |y_1 - y_2|, \text{ with } p,q \in \mathbb{R}, p = (x_1, y_1), q = (x_2, y_2).$$

Figure 44(a) shows the distance transform of the toy-example(see Figure 20).
After that, the gradient image of $T$ is calcutated. This is defined for a function of two variables $F(x,y)$ as:

$$\nabla F = \frac{\delta F}{\delta x} + \frac{\delta F}{\delta y}.$$

Literally speaking, the gradient image of $F$ is a collection of vectors that are pointing in the direction of increasing values of $F$:

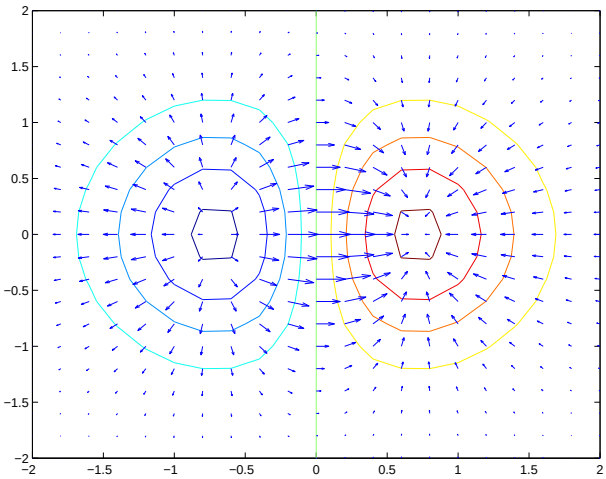



Figure 43: Example image for the gradient calculations.

For each pixel $p$ that has been marked to be a point where two cells are overlapping and need to be divorced, the gradient vector of p is pointing in the right splitting direction. The algorithm follows the directions of the gradient vectors pixel by pixel until a local maximum of the distance transform of the current object is reached. After that, the negative vector directions are followed. In each step, the pixel is set to the background value.
The marked positions (see Figure 44(b)) are initialising the splitting algorithm as described above. The result is presented in Figure 44(c).
These results can be optically improved by applying the smoothing algorithm once more.

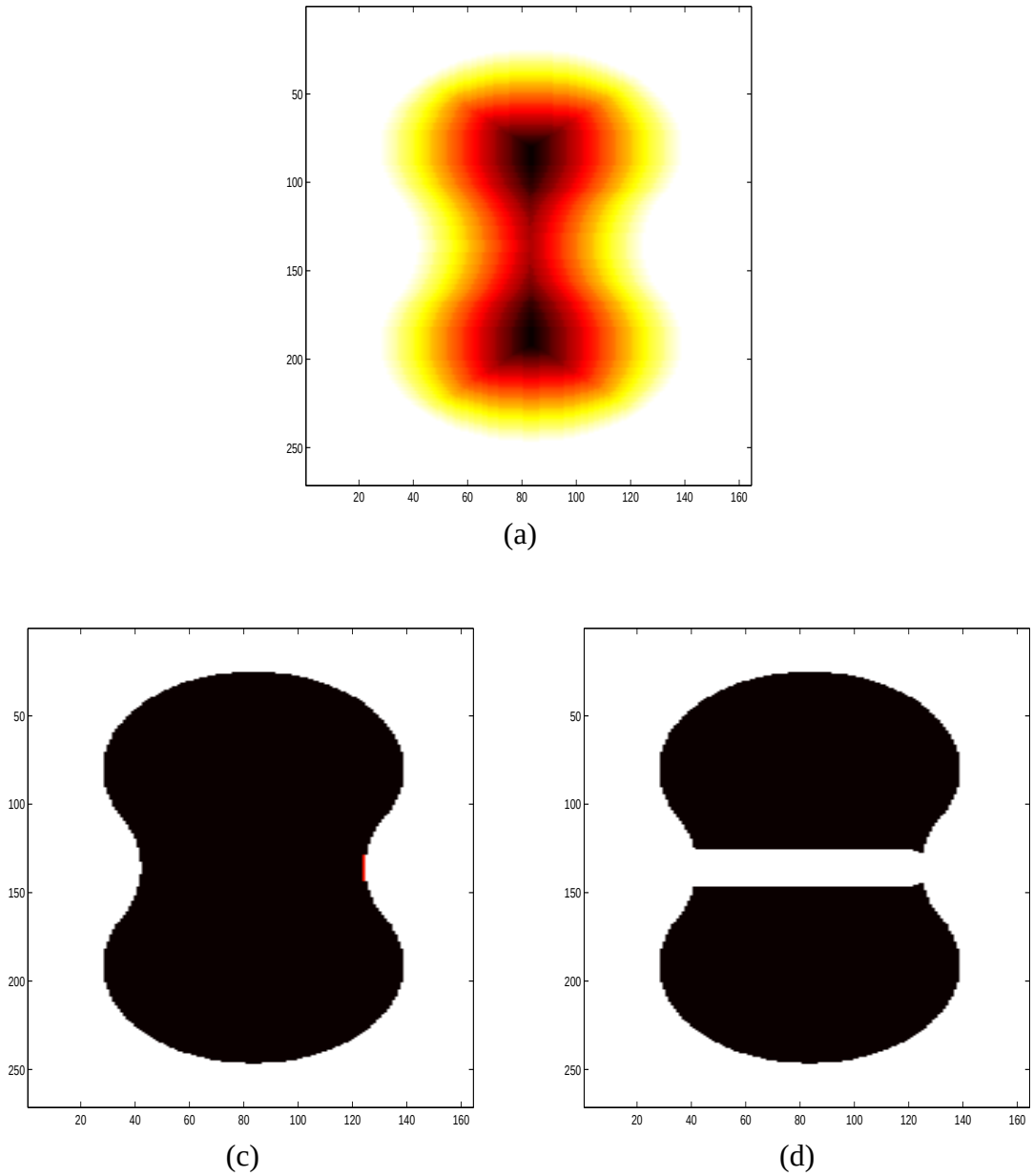



Figure 44: The different steps of the splitting algorithm. (a): Distance transform; (b): Marked positions; (c): Divided object.

## 5.6 Final Results

Finally, the fast local curvature estimation is applied again on the complete cell dataset. After that, convexity calculations are done. Because of the lower z-resolution of the confocal laser scanning results, it turned out to be sufficient to do so for each x/y-slice of the data volume. This is done for those pixels only, where the results of the curvature estimation exceeds a certain threshold. Therefore, the procedure becomes much faster then calculating convexity for all dataset entries. Afterwards, only the concave object positions remain. Here, it is also possible to choose another threshold for the degree of concavity. At this point, all positions, where two different cells are overlapping are localized and the splitting algorithm can be applied. The smoothing algorithm that has been used at the beginning is run again on the data volume.
The results of each single processing step are illustrated in Figure 45.

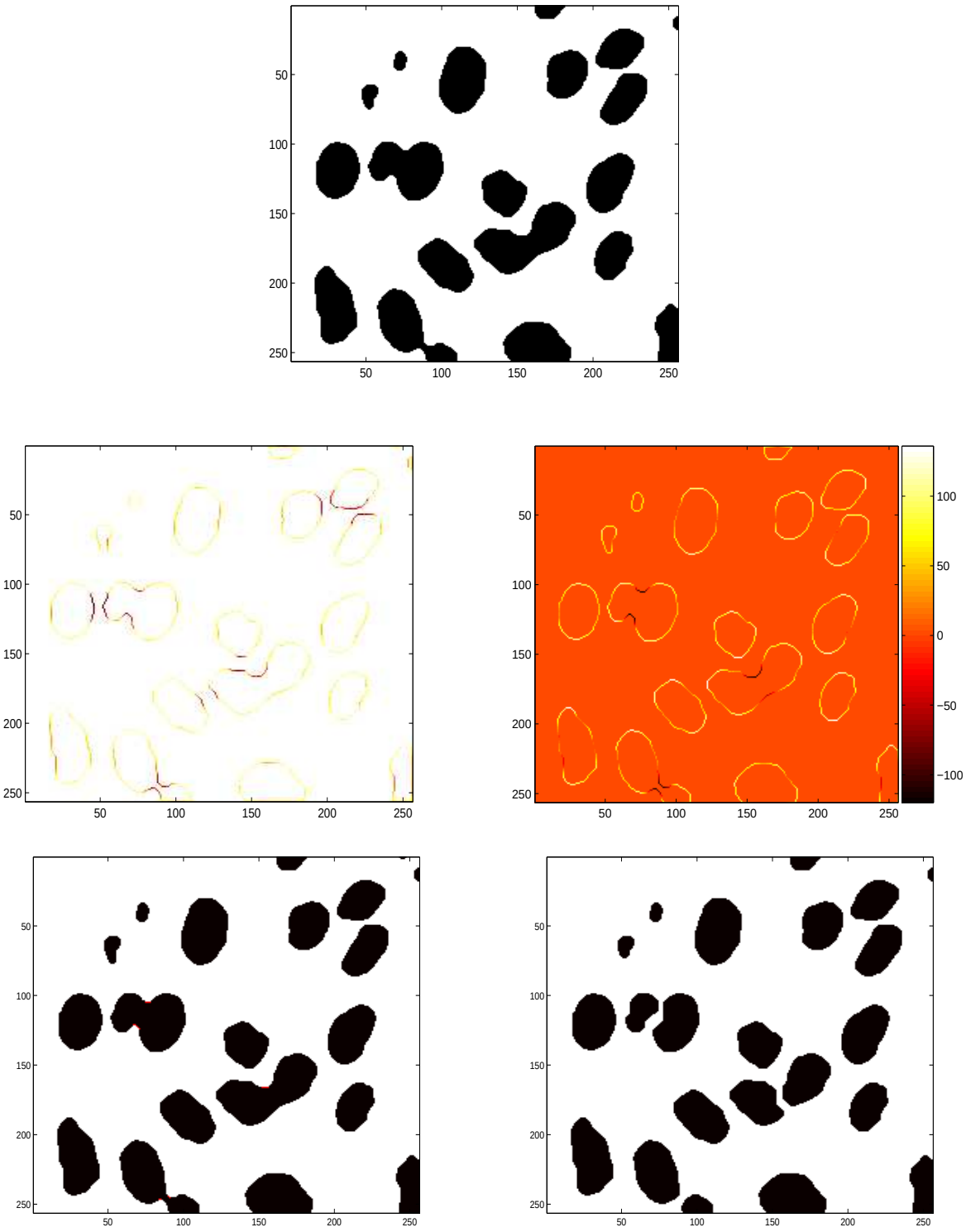



Figure 45: Results of the cell dataset.

# 6 Conclusions

In this work, all basics were formed to use associative Markov networks for the segmentation of biological structures in 3D volumetric data. A random Markov field was constructed, including different types of feature extraction methods. A new feature for local curvature estimation in 3D volumetric data was found, which is invariant under monotonic gray-scale changes and has also shown to be robust towards noise. This feature can be computed for all positions in the volume at once using a non-iterative and thus very efficient algorithm. Also it has been shown that this new method is capable of detecting regions, where two adjacent objects lie so close to each other that established methods would classify them as one. It works well for different types of object shapes in 2D and 3D space. Correct separation has been supplied in this work as well.
The next step would be a generalization on multiple datasets, using the proposed learning algorithm for AMNs. Markov chain Monte Carlo methods were introduced to use sampling from probability distributions based on the construction of Markov chains. It has been shown, that with those methods a solution for the optimization problem is quite attainable.

# A Softwaretools

## Blitz++

Blitz++ is a C++ class library for scientific computing. It uses template techniques to achieve high performance. Blitz++ provides dense arrays and vectors, random number generators, and small vectors, which are suitable for discrete image processing.[4]

## Matlab

Matlab is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numeric computation.[5]

## NetCDF

The Unidata network Common Data Form (netCDF) is a platform independent data format. It is suitable for a homogeneous representation of scientific data. The belonging interface and the implementation are freely distributed. NetCDF is optimized for n-dimensional array-structured data. It is also appropreate for saving images and volume data. These can been saved in addition to any other informations about the data. The current netCDF software provides C, Fortran-77, Fortran-90, and C++ interfaces for applications and data.[6]

## ParaView

ParaView is an open-source, multi-platform visualization application. It is designed for large data sets and supports distributed computation models.[7]

# B Abbreviations

| | |
|---|---|
| AMN | Associative Markov Networks |
| CAM | Chorioallantoic Membrane Probes |
| FFT | Fast Fourier Transformation |
| GRF | Gibbs Random Field |
| LSM | Laser Scanning Microscope |
| MAP | Maximum A-Posteriori |
| MCMC | Markov Chain Monte Carlo |
| MCMCM | Markov Chain Monte Carlo Methods |
| MRF | Markov Random Field |
| SVM | Support Vector Machine |

[4] http://www.oonumerics.org/blitz/whatis.html.
[5] http://www.mathworks.com/products/matlab/.
[6] http://www.unidata.ucar.edu/packages/netcdf/.
[7] http://www.paraview.org/HTML/Index.html.

# References


[1] D. Anguelov et. Al. Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data. In: CVPR 2005.

[2] B. Taskar, V. Chatalbashev and D. Koller. Learning associative Markov Networks. In: ICML 2004.

[3] J. Fehr, O. Ronneberger, O. Kurz and H. Burkhardt. Self-learning Segmentation and Classification of Cell-Nuclei in 3D Volumetric Data using Voxel-Wise Gray Scale Invariants. In: W. Kropatsch, R. Sablatnig and A. Hanbury (Eds): DAGM 2005, LNCS 3663, pp 377-384.

[4] J. Fehr. Segmentation and Classification of Cell Nuclei in Tissue Slices using Voxel-Wise Gray-Scale Invariants. Diploma thesis. November 2004.

[5] http://www.zeiss.de/lsm.

[6] http://www.mikroskopie.de/kurse/fluoreszenz/elektron.html.

[7] http://www.sinnesphysiologie.de/methoden/fluo/fluomikl.htm.

[8] J. Fehr, O. Ronneberger, M. Reisert, J. Schulz, T. Schmidt and H. Burkhardt. Invariance via Group-Integration: Designing Features for 3D Volumetric Data in Biomedical Image Analysis.

[9] J. Fehr, C. Sauer, H. Kurz, O. Ronneberger and H. Burkhardt. Identifikation von Zellen in intaktem Gewebe; Segmentierung und Klassifikation von Zellkernen in 3D Volumendaten mittels voxel-basierter Grauwertinvarianten.

[10] G. Winkler. Image Analysis, Random Fields and Markov Chain Monte Carlo Methods, A Mathematical Introduction. Springer 1995,2003.

[11] B. Vidakovic. Handout 16. http://www2.isye.gatech.edu/ brani/isyebayes/handouts.html.

[12] T. Fan, G. Medioni and R. Nevatia, Description of Surfaces From Range Data Using Curvature Properties, in Proc. Comput. Vision Patt. Recogn., pp. 86-91, 1986.

[13] R. Klette. Algorithms for Picture Analysis. Albert-Ludwigs-University Freiburg 2005.

# List of Figures

# List of Tables