computer graphics & vision

Inffeldgasse 16/2, A-8010 Graz

www.icg.tu-graz.ac.at

TUG

Semantic Segmentation with Patch Priors

Master's Thesis

Christian Rauer, BSc

Advisor:
Michael Donoser, PhD

Institute for Computer Graphics and Vision
Graz University of Technology, Austria

Technical Report
ICG–TR–THESIS
Graz, January 21, 2013

contact: Christian Rauer rauer@student.tugraz.at

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

. . . . . .
(date)

. . . . . .
(signature)

Abstract

The topic of semantic segmentation is an important part of many applications in computer vision. This master's thesis presents a method for the incorporation of prior knowledge on the patch level in the inference process of a standard semantic segmentation pipeline. The presented approach uses a simple yet efficient method for the learning of prior knowledge by building a histogram of the occurrences of all image patches of a certain size in a set of training images. The implemented semantic segmentation framework extends the state-of-the-art approach by iteratively modifying the output of a graph cuts framework and reapplying the graph cuts regularization to the updated intermediate results obtained in this way. The previously constructed patch prior databases are utilized to determine the likelihood of patches in the intermediate results under the learned patch prior. In this way impossible patch configurations can be completely ruled out, and image patches that only occur infrequently in the training images are assigned a low likelihood. As a result, incorrectly labeled areas that can otherwise not be eliminated by the state-of-the-art approach can be dealt with.

This thesis starts out with a survey of related work on the topic of segmentation with a special emphasis on semantic segmentation. It also deals with some of the concepts that are used as basis for these works. This is followed by an introduction of the datasets used in the evaluation phase and a detailed description of the implemented method. A discussion of the conducted experiments and the results and conclusions drawn from them demonstrates the usability of the presented method for the task of semantic segmentation.

Keywords: *semantic segmentation, patch prior, graph cuts*

Kurzfassung

Das Thema der Semantischen Segmentierung ist ein wichtiger Teil vieler Anwendungen der Computer Vision. Diese Diplomarbeit präsentiert eine Methode für die Inklusion von Prior-Wissen (Vorwissen), das auf dem Patch-Level (Bereich um ein Pixel) ermittelt wird, im Inferenz-Prozess der Standardvorgehensweise bei der Semantischen Segmentierung. Das präsentierte Verfahren verwendet eine einfache aber effiziente Methode für das Lernen des Prior-Wissens durch die Erzeugung eines Histogramms des Auftretens aller Bild-Patches einer bestimmten Größe in einem Set von Trainingsbildern.
Das implementierte Framework zur Semantischen Segmentierung erweitert die Vorgehensweise, die dem derzeitigen Stand der Technik entspricht, durch die iterative Aktualisierung des Outputs eines Graph Cuts-Frameworks und die nochmalige Anwendung der Graph Cuts Methode auf die so erhaltenen Zwischenergebnisse. Die zuvor errechneten Patch Prior-Datenbanken werden genutzt um die Wahrscheinlichkeit von Patches in den Zwischenergebnissen, unter der Einschränkung des gelernten Patch Priors, zu ermitteln. Auf diese Weise können unmögliche Patch-Konfigurationen komplett ausgeschlossen werden und den Patches, die nur selten in den Trainingsbildern vorkommen, wird eine niedrige Wahrscheinlichkeit zugewiesen. Damit können falsch bezeichnete Bereiche, die auf andere Weise nicht von der Standardmethode kompensiert werden können, behandelt werden.
Diese Diplomarbeit beginnt mit einer Übersicht über verwandte Arbeiten die Segmentierung thematisieren, mit einem besonderen Schwerpunkt auf Semantischer Segmentierung. Es werden ebenfalls einige Konzepte behandelt, die als Basis für diese Arbeiten dienen. Darauf folgt eine Vorstellung der verwendeten Datensets, die in der Evaluierungsphase verwendet wurden, und eine detaillierte Beschreibung der implementierten Methode. Eine Diskussion der durchgeführten Experimente und die Ergebnisse und Schlussfolgerungen, die daraus gezogen werden können, demonstriert die Anwendbarkeit der präsentierten Methode für die Semantische Segmentierung.

Schlüsselwörter: *Semantische Segmentierung, Patch Prior, Graph Cuts*

Danksagung

An dieser Stelle möchte ich besonders meinen Eltern danken, die mir mein Studium durch ihre Unterstützung und ihre Geduld ermöglicht haben. Auch bei meiner Schwester Katharina und meinem Bruder Johannes will ich mich bedanken, die mir durch ihren Fleiß eine Inspiration waren und sind.

Ebenso möchte ich meinen Freunden und Studienkollegen, die mich während der vergangen Jahre begleitet haben, meinen Dank aussprechen.

Außerdem bedanke ich mich bei DI Dr. Michael Donoser für die hervorragende Betreuung während dieser Diplomarbeit.

Zum Schluss möchte ich auch noch meiner Schwester Katharina und meiner Freundin Eva für das Korrekturlesen meiner Diplomarbeit danken.

Contents

List of Figures

List of Tables

1 Introduction

Semantic segmentation is a core problem in computer vision. The goal of this task is to assign a label to each pixel of an input image, specifying the semantic class to which the pixel belongs.
This master's thesis describes a method for the improvement of a common semantic segmentation process by incorporating prior knowledge at the patch level in a standard semantic segmentation pipeline.
This section starts out by giving a motivation for the task of semantic segmentation in general and the presented method in particular in Chapter 1.1. Then the main contribution of this master's thesis to the topic of semantic segmentation is discussed in Chapter 1.2. Finally, this section is concluded by an overview of the individual parts of the following thesis with a short synopsis of each section in Chapter 1.3.

1.1 Motivation

Segmentation in terms of computer vision is the task of dividing an input image into a set of non-overlapping regions. In the case of semantic segmentation the non-overlapping segments of the input image are additionally labeled as belonging to a class from a finite set of semantic categories that correspond to areas and objects in the real world like 'grass', 'sky', or 'person'. Figure 1 shows an example image and the corresponding groundtruth consisting of the two semantic categories 'grass' (green) and 'cow' (blue).

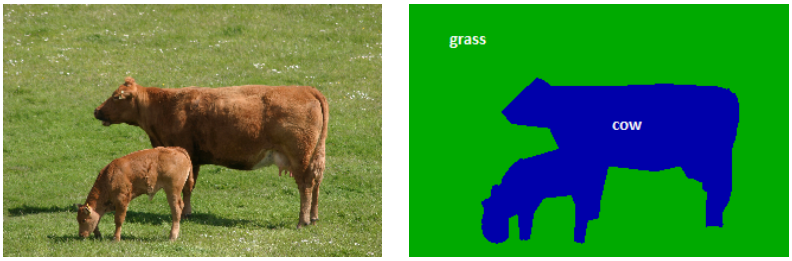

Figure 1: Example image and corresponding groundtruth with the two semantic classes 'grass' and 'cow'.

The semantic segmentation problem can be posed as a minimization problem where the relationships between pixels in the image are represented by energies. This energy minimization problem can in turn be efficiently solved by representing these energies with nodes and edges in a graph. The final solution to the optimization problem can then be found with graph-based techniques like the graph cuts regularization.

There are many methods following this approach, some of which are discussed in Section 2 which deals with related work. The current state-of-the-art approach consists of using a data structure to represent the per pixel likelihood for belonging to each of the possible semantic categories and applying graph cuts regularization on that data structure. Figure 2 shows the maximum likelihood estimation (MLE) on this data structure on the left. The MLE determines the most likely class label for each pixel according to the precalculated data structure, and corresponds to a rough initial segmentation. The result of the application of the graph cuts regularization is depicted on the right of Figure 2.

Figure 2: Maximum likelihood estimation and result of graph cuts regularization on representation of pixel relationships.

The two semantic segmentations, resulting from the MLE and the graph cuts regularization in Figure 2, show slight differences in areas that correspond to a transition between the two semantic classes. The actually measurable difference in terms of segmentation accuracy gets evident when the two segmentations are evaluated with an evaluation framework that compares the segmentation results to the desired result represented by the annotated groundtruth that is depicted on the right in Figure 1. The results of this evaluation for the MLE and the graph cuts regularization on the data structure representing the inter-pixel relationships are shown in Table 1.

Table 1: Segmentation accuracies for the example in Figure 2.

	grass	**cow**	**class accuracy**	**pixel accuracy**
MLE	92.48%	95.65%	94.07%	93.30%
Graph Cuts	93.00%	95.87%	94.43%	93.74%

The numbers in Table 1 show that the graph cuts regularization leads to a slight increase in class accuracy compared to the MLE for both the 'grass' and the 'cow' class, which in turn also results in increases in average class accuracy and average pixel accuracy.

Even though the application of the graph cuts regularization usually leads to an improvement in terms of segmentation accuracy, this standard method can not compensate for errors that are based on a flawed representation of the inter-pixel relationships that is derived from the pre-calculation step.
Figure 3 shows an example of a segmentation containing areas with incorrect semantic classes, some of which can not be eliminated by the graph cuts approach.

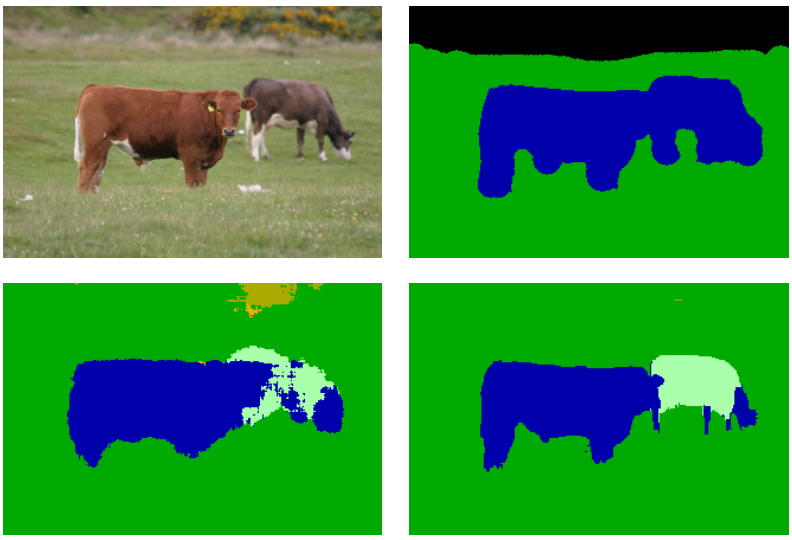

Figure 3: Example image and corresponding groundtruth (top), MLE and result of graph cuts regularization on representation of pixel relationships with incorrect semantic classes (bottom).

This example shows that the graph cuts regularization can eliminate some of the incorrectly labeled areas, like the brown and orange sections at the top in the bottom left image of Figure 3. It also shows that the graph cuts approach reaches its limits if the erroneous area is too large, like the light green area on the right cow that corresponds to the semantic class 'dog', which even gets enlarged by the graph cuts regularization. The problem is based on the fact that the graph cuts approach does not take the likelihood of the co-occurrence of semantic categories into account.
The examples in this chapter show that the state-of-the-art approach of using graph cuts regularization leads to improvements in segmentation accuracy. The method presented in this master's thesis aims at additionally improving the classification accuracy by analyzing and applying previously learned prior knowledge to avoid impossible or unlikely constellations of semantic categories.

1.2 Contribution

The main contribution of this master's thesis to the topic of semantic segmentation is that the method presented here uses a comparatively simple approach to extend the basis of the state-of-the-art approaches and is at the same time also able to achieve results that can compete with more sophisticated methods.
As the title already suggests, the main idea is to incorporate prior knowledge on the patch level. This prior knowledge is learned from a set of input images that consists of the manually annotated semantic segmentation groundtruth for the respective original images. The prior knowledge is learned by building a patch prior database that holds all patches occurring in the training images and their respective numbers of occurrences. This histogram of all occurring patches gives an insight about the likelihood of the occurrence of image patches in the input images according to the prior learned from the training images.
Figure 4 shows examples of different patches of size 3×3 extracted at homogenous locations and at the transition between areas of two different semantic classes.

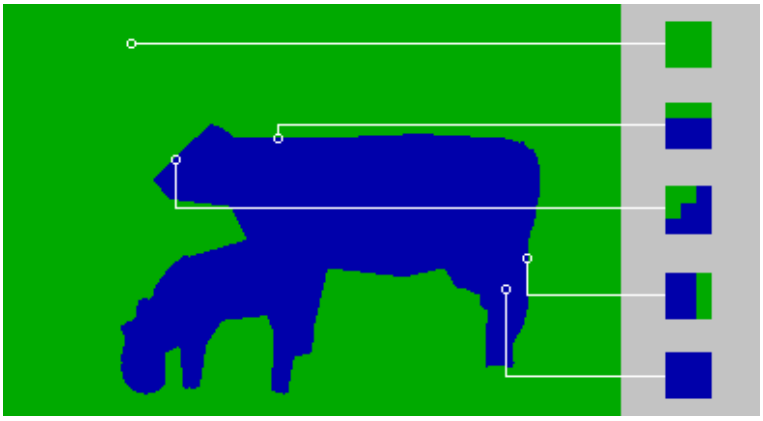

Figure 4: Example of patches extracted from a training image (patch size 3×3).

Different databases are built for patch sizes varying between 3×3 and 15×15. The image patches are extracted at each pixel location in all training images and the number of their occurrences is stored in the resulting databases.
The following example, that uses a database constructed from four training images, demonstrates the approach taken in this master's thesis to learn the prior knowledge and apply it to avoid unlikely configurations of semantic categories.

Figure 5 shows the four training images containing cows on grass and dogs on grass that are used in this example.

Figure 5: Training images containing cows on grass and dogs on grass.

Figure 6 shows the 50 most frequent patches of size 3×3 along with the number of their occurrences that appear in the database built for the training images in Figure 5.

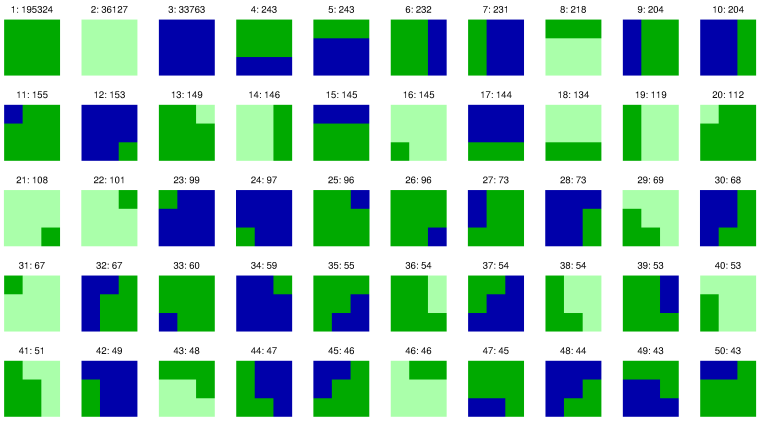

Figure 6: 50 patches with the highest number of occurrences constructed from the training images from Figure 5 (patch size 3×3).

The database excerpt presented in Figure 6 shows that for this example the homogenous patches that only consist of 'grass', 'cow', or 'dog' pixels have the highest number of occurrences. The other patches consist of transitions between the semantic classes 'grass' and 'cow' or 'grass' and 'dog'.
While it is not impossible that an input image could contain a scene where cows and dogs are photographed on grass at the same time, where areas of 'cow' and 'dog' pixels touch, such a configuration is unlikely (actually 0, which means impossible) under the prior learned from the training images in Figure 5, since it does not contain any such patches.
The presented method computes a probability for each of the possible semantic classes at every pixel location, according to the learned prior and applies the graph cuts regularization on the data structure that is modified in this way. These steps are repeated iteratively until convergence.

Figure 7 shows the result that is obtained with the presented method. The previously incorrectly labeled areas at the bottom of Figure 3, that corresponded to the semantic class 'dog', are now correctly labeled as belonging to class 'cow'.

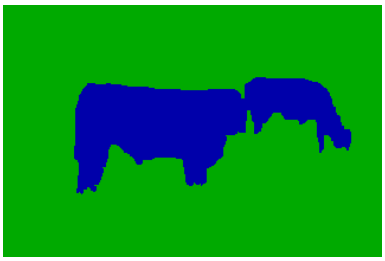

Figure 7: Segmentation result for the input image from Figure 3 calculated by the implemented method.

Table 2 presents the segmentation accuracies that were determined in the comparison with the annotated groundtruth for the segmentation results of the MLE and the graph cuts regularization presented at the bottom of Figure 3 and the segmentation result that could be achieved with the implemented method as shown in Figure 7.

Table 2: Segmentation accuracies for the segmentation results presented in Figures 3 and 7.

	grass	**cow**	**class accuracy**	**pixel accuracy**
MLE	96.02%	71.66%	83.84%	89.12%
Graph Cuts	99.77%	63.40%	81.58%	89.46%
Method	99.85%	76.96%	88.41%	93.37%

Table 2 shows that the flawed representation of the inter-pixel relationships, that leads to an incorrectly labeled area of pixels from class 'dog', yields a comparatively low class accuracy for the class 'cow'. The enlargement of this incorrectly labeled area by the graph cuts regularization, presented at the bottom right in Figure 3, leads to a further decline in the class accuracy for this class. The segmentation accuracy for the class 'grass' increases by over 3% which results in a slight gain in pixel accuracy for the graph cuts regularization. The segmentation obtained by the method implemented for this master's thesis can eliminate the incorrectly labeled pixels which results in an improvement of the class accuracy for class 'cow' which also leads to an increase of the average pixel accuracy and the average class accuracy.

Summing up, the contribution of the method presented in this master's thesis consists of the development of an efficient procedure for the learning and application of prior knowledge at the semantic level that is extracted at the patch level. The learned prior knowledge is stored in a database in a space- and time-efficient manner that allows fast and easy access for writing in the learning stage and reading during the application. The databases built in this way are then applied in an iterative process that gradually refines the semantic segmentation results, starting at a rough initial segmentation. The goal is to avoid impossible and unlikely configurations of semantic categories and in this way, additionally to an improvement of the segmentation accuracy, also leading to an improvement in terms of classification accuracy.

1.3 Synopsis

This chapter gives a brief overview of this master's thesis by shortly summarizing each of the following sections.

Section 2 gives an overview of previous work on the topic of segmentation and semantic segmentation in particular. It also summarizes work on other topics involved in these methods like image features, methods for the modeling of learned training data, and inference methods.

Section 3 discusses the state-of-the-art approach of considering the semantic segmentation problem as an energy optimization problem and the means of representing the involved energies.

Section 4 presents the graph cuts regularization approach that is used to iteratively update the semantic segmentation.

Section 5 discusses the details of the incorporation of the prior knowledge on the semantic level.

Section 6 presents the method that was developed in the course of this master's thesis with a detailed discussion of the contributions to the topic of semantic segmentation.

Section 7 introduces the datasets used in the experiments and evaluations that are conducted with the implemented method.

Section 8 discusses the databases that were built during the learning stage. Some interesting aspects of the databases that come with different patch sizes are discussed and illustrated with tables and diagrams.

Section 9 presents the results that could be achieved with the implemented method in the form of tables and diagrams along with corresponding interpretations of the outcomes and comparisons to the results that were reported by the current state-of-the-art semantic segmentation methods.

Section 10 offers a final discussion of the achieved results and the conclusions that can be drawn from them. It also gives an outlook of possible improvements that could be achieved by further enhancing the presented method.

2 Related Work

This section gives an overview of related work on the underlying concepts involved in this master's thesis. It starts out by describing a number of early and recent efforts on the subject of segmentation in Chapter 2.1. A special emphasis is of course put on the topic of semantic segmentation which is covered in Chapter 2.2. A topic that is relevant in correlation with semantic segmentation is the concept of conditional random fields. Related work on this topic is discussed in Chapter 2.3. Chapter 2.4 covers prior work on the topic of graph cuts and its significance in the context of semantic segmentation. The section is concluded by a review of previous work on the subject of patch priors that are used to learn prior knowledge in Chapter 2.5.

2.1 Segmentation

Segmentation in the context of computer vision is the challenge of dividing an input image into a set of non-overlapping regions. The goal is to split the image into parts that have a strong correlation with objects or areas of the real world that are contained in the image [43]. This processing step leads to a data representation that is more meaningful and/or more efficient for further analysis [44]. The resulting simple data representation can then be used as basis in other computer vision tasks.
Numerous applications are based on segmentation results, for example object detection, where the challenge is to differentiate between objects and the image background, or edge detection which finds the borders of the objects in an image. One very important application, that the master's thesis at hand deals with, is the task of semantic segmentation, where the goal is to assign labels from a pre-defined set of object categories to the corresponding areas in the input image.
The first methods for image segmentation were developed in the late 1970s and early 1980s. These first efforts used low-level techniques such as thresholding of intensity values [34], region growing [10], region splitting [35], and watersheds [7, 6]. The focus of these methods lies mainly on the similarity of intensity values in the images to be segmented. While there still exists recent work dealing with image segmentation methods using these techniques of thresholding [36, 37], watersheds [38], region growing [30], region splitting [2], and other region based methods [4, 3] higher-level segmentation methods like semantic segmentation usually utilize special image features and parametric methods to achieve their goal.

2.2 Semantic Segmentation

Semantic image segmentation is the task of assigning a label, according to an underlying category, to each pixel of the image to be segmented, such that pixels that are labeled identically represent identical parts of the image.
Most semantic image segmentation methods have in common that they use some kind of image features to extract information from the input images. In the case of parametric methods this information is learned on a set of training images, and the learned knowledge is then applied to perform semantic segmentation on a set of test images. This chapter gives a short overview of the image features used in this process and then presents some of the most important recent semantic segmentation methods.

2.2.1 Features

The features used to retrieve information from the input images range from the scale invariant feature transform (SIFT) descriptors [32, 33] as used by Vezhnevets et al. [49], through Histogram of Gradients (HoG) [39], Gaussian mixture models (GMM) [18], combinations of different types of filter responses like first and second order derivatives of Gaussians, Laplacians, and Gabors [50], covariance descriptors [22], to color and texture features. In some cases a combination of some of the above features is used.
One kind of feature that has gained special interest in recent work are textons [29]. Textons are capable of modeling an object's shape, appearance, and context [41] by performing a convolution with a multi-dimensional filter bank. The 17-dimensional vector, that a texton consists of, captures filter responses with a combination of Gaussians, x and y derivatives of Gaussians, and Laplacians of Gaussians. The filters are applied to all three color channels and the luminance channel in the CIELab color space [42]. These textons are clustered with an unsupervised Euclidean-distance k-means clustering algorithm. Each pixel in each image is then assigned to the nearest cluster center, thus producing a texton map for each image. The textons are used for a novel texture-layout filter feature which is a pair of an image region and a texton.

2.2.2 Methods

The texton feature was first used and developed by Shotton et al. for the TextonBoost system [42]. With textons they introduce an approach for the learning of a discriminative model of object classes by efficiently incorporating information from multiple levels of the input image. The method uses conditional random fields (CRF) to incorporate the texture, layout, color,

location, and edge cues in a single unified model. The resulting function for the conditional probability of the class labels contains terms for these edge cues which can be included in the function or not, to turn certain features on and off. The optimal labeling is found by applying as inference method on the CRF the α-expansion graph cuts algorithm [48, 12], which is discussed in more detail in Chapter 4.3.
The paper of Shotton et al. shows that the improvements that come with the use of this new full CRF model, while being numerically small from the view of segmentation accuracy, lead to an improved segmentation.

Ladický et al. [27] follow a similar approach. They use a class of global potentials defined over all variables in a CRF, and they also use graph cuts algorithms for optimization.
They define a set of constraints for the cost function over the CRF that have to be met. The global energy formulates a co-occurrence that allows estimation of the segmentation using all the data directly by minimizing a single cost function rather than any sort of two stage process. The invariance constraint states that the co-occurrence cost should depend only on the labels present in an image. It should be invariant to the number and location of pixels that object occupies. The efficiency constraint demands that inference must be tractable. The use of co-occurrence should not be the bottleneck preventing inference. Finally, the parsimony constraint states that the cost should follow the principle of parsimony which means that if several solutions are almost equally likely, the solution that can describe the image using the fewest distinct labels should be chosen.
The complexity for solving the labeling problem with multiple labels is very high since the number of additional edges in the graph grows linearly with the number of variables in the graph. For optimization so called move making algorithms are utilized. These algorithms project the problem into a subspace in which the sub-problem can be solved efficiently. For this method in particular α-β swap and α-expansion moves [48, 12] are used. The details of the graph cuts algorithms for multiple labels are discussed in Chapter 4.3.

The approach of Krähenbühl and Koltun [25] is similar to the previous two methods in that the CRF model is used. The authors use fully connected CRF models defined on the complete set of pixels in an image. Pairwise potentials are established on all pairs of pixels in the image which leads to a refined segmentation and labeling. The resulting graphs contain tens of thousands of nodes and billions of edges even on low resolution images which makes a traditional inference algorithm like the graph cuts regularization approach impractical.

Instead they propose an efficient approximate inference algorithm for fully connected CRF models in which the pairwise edge potentials are defined by a linear combination of Gaussian kernels. The approximation is iteratively optimized through a series of message passing steps, each of which updates a single variable by aggregating information from all other variables. The update of all variables in a fully connected CRF can be performed by Gaussian filtering in feature space which allows the reduction of the computational complexity of message passing from quadratic to linear in the number of variables.
The unary potentials used in the implementation are derived from TextonBoost [42]. The 17-dimensional filter banks are extended by adding color, histogram of oriented gradients (HOG), and pixel location features.

He et al. [19] propose an approach to include contextual features for the labeling of images in which each pixel is assigned to one of a finite set of labels. The features are incorporated into a probabilistic framework where the outputs of several components are combined. The result is a combination of a segmentation of the image and a recognition of each segment as one of the object classes.
This method again uses a CRF approach. Features are considered on local, regional, and global scales and are integrated into a multiscale conditional random field (mCRF). The mCRF framework is instantiated with three separate components operating at three different scales: a local classifier, regional features, and global features. At the local level, pixels are classified using a statistical classifier such as a neural network. Regional label features are used to represent local geometric relationships between objects such as edges or corners. These features prevent impossible label combinations such as a border of ground above sky and are obtained by a division of the label field for the whole image into overlapping regions. Each global feature has as its domain the label field for the whole image.

Another similar approach is taken by Kluckner et al. [22]. Again the CRF model is used, and additionally to other image features, height information is integrated to semantically segment aerial images. The authors present a technique to obtain accurate semantic segmentation on the pixel level, capable of integrating appearance cues such as color, edge responses, and height information. The paper uses covariance descriptors which provide a low-dimensional feature representation that can simply integrate multiple feature channels such as color, filter response, or height information and can also exploit the correlation between them. Since the space of the covariance matrices is non-Euclidean, machine learning methods cannot be used directly.

To represent the individual covariance matrices directly in Euclidean vector space the concept of Sigma Points is used. The final feature representation is obtained by a concatenation of these Sigma Points, and it can be used in a multi-class randomized forest classifier framework. Additionally semantic contextual knowledge is included using a CRF formulation.

The previous methods discussed had all in common that they used a parametric approach, which means that images can only be semantically segmented after first performing a learning process on training data.
Liu et al. [31] present a non-parametric approach for object recognition and scene parsing via label transfer.
The main idea of the presented system is recognition by matching. The visual objects in the input images are matched to images in the database which are annotated with category labels. These labels are transferred to the input image to achieve a semantic segmentation.
The system pipeline contains three main steps. First scene retrieval techniques are used to find a set of nearest neighbors, that share a similar scene configuration with the input image, from a large database containing annotated images. For that purpose the system uses a combination of k-NN and ϵ-NN to find candidates in the training database. Then a dense scene correspondence between the input image and each of the retrieved nearest neighbors with the top matching scores is established using the SIFT flow algorithm. Finally, based on this correspondence, the best matches from the database are warped to match the input image according to the estimated dense correspondence from the previous step and the annotated labels are transferred to the query image by resolving multiple labelings and imposing spatial smoothness under a Markov random field (MRF) model that incorporates multiple cues such as likelihood, prior, and spatial smoothness.

Schnitman et al. [40] present another non-parametric approach that uses only a single training image. They construct a non-parametric representation of the segmentation of this training image by selecting patch-based representatives inside each labeled region of the training image. These representatives quantify the degree of resemblance between small regions in the input image and the labeled regions of the training image. The input image is then partitioned into small homogenous fragments and a possible labeling is found with a voting procedure under the presumption that homogenous regions always belong to the same semantic part of the image. Then the graph cuts approach is used to label each fragment such that a labeling is achieved that is globally optimal.

Another method that uses the conditional random field model was developed by Toyoda and Hasegawa [46]. They present a framework that explicitly models local and global information in a conditional random field. The incorporation of the global information resolves local ambiguities. The method characterizes scenes by global image features and generates scene-based top-down information or prior knowledge about the scene. The framework derives the global information from top-down information as a form of a predicted spatial layout and category compatibility. A predicted layout for example states that a scene probably includes a road with a car in the center of the image. The category compatibility might state that a polar bear probably will not appear in the same scene as a hippopotamus.
The labeling of scenes is performed at the superpixel level where the superpixels correspond to the sites in the proposed model. The label of each individual pixel is determined by the label of the superpixel it belongs to.
While the presented method is very similar to others that are more up to date, this paper shows how the CRF is built from the local and global image features in a very detailed way, and how these features are extracted from the image.

Semantic segmentation is again discussed in more detail in Section 3 that includes the definition of the semantic segmentation problem as an energy minimization problem and how the energies involved can be represented by data structures called potentials.

2.3 Conditional Random Field

The previous chapter shows that many of the recent papers on the topic of semantic image segmentation utilize some form of the conditional random field model as a means to efficiently model the information that is learned with parametric methods.
A random field is a stochastic process that takes values in a Euclidean space, and it is defined over a parameter space of a dimensionality of at least one [1]. In the case of the application of random fields in computer vision tasks, each image pixel is a random variable in the random field model, ranging over the possible labels that can be assigned to each pixel.
A useful extension of the random field model, that can easily be applied to tasks posed in computer vision, is the Markov random field (MRF) model [45]. MRFs are used to model the joint probability of the image and its corresponding labels. One problem that occurs with the use of MRFs is that due to the high complexity of inference and parameter estimation, only the local relationships of neighboring pixels are incorporated in the model which

makes it inefficient for the capturing of interactions of pixels that are further apart [19].
This problem can be avoided by directly modeling the conditional probability of labels given images [19] in a conditional random field (CRF) [28]. This method was first developed by Lafferty et al. and was originally intended for the segmentation and labeling of text sequences. The authors present a sequence modeling framework for building probabilistic models to segment and label sequence data. A conditional model specifies the probabilities of possible label sequences given an observation sequence. The CRF has a single exponential model for the joint probability of the entire sequence of labels in the given observation sequence. A CRF can be thought of as a finite state model with unnormalized transition probabilities. CRFs assign a well-defined probability distribution over possible labelings trained by maximum likelihood or maximum a posteriori (MAP) estimation. The authors show that their novel approach is an alternative that can outperform hidden Markov models (HMM) and maximum entropy Markov models (MEMM).
This method that was originally developed for the segmentation and labeling of 1-D text sequences can be generalized for the solution of the same tasks in the case of 2-D image data for computer vision tasks. There exist further extensions like multiscale CRFs [19] or hierarchical CRFs [26] but they all have in common that the different random fields capture the relationships between the pixels of images. There exist different methods for the inference of a solution from such a random field model. One very efficient method is the graph cuts regularization approach that is discussed in the next chapter.

2.4 Graph Cuts

To infer a solution for the different random field models, and for the conditional random fields model in particular, different approaches can be applied. Among other methods like loopy belief propagation (LBP) [52] or iterated conditional modes (ICM) [5], the graph cuts regularization approach is one of the most efficient and most commonly used [45].

Greig et al. [17] were the first to apply the graph cuts approach in computer vision. They show how the maximum a posteriori (MAP) estimate of the true scene can be found exactly for binary images. This is achieved by determining the maximum flow in a capacitated network by reformulating the problem as a minimum cut problem. The results achieved with the new method are superior to simulated annealing. The authors state that the simple network flow algorithm does not extend to multicolor scenes in an obvious way since they could not find a corresponding network formulation.

Wu and Leahy [51] present another early approach to utilize the graph cuts approach in computer vision. They present a graph theoretic method for data clustering and its application to the image segmentation problem.
The data to be clustered is represented by an undirected graph where each node corresponds to a data point and an edge connects two nodes if the corresponding data points are neighbors according to a given neighborhood system. Each edge is assigned a flow capacity that reflects the feature similarity between the pair of nodes. Clustering is achieved by removing edges corresponding to minimum cuts between node pairs from the graph, to form mutually exclusive sub-graphs. The minimum cuts of the graph are computed from a flow and cut equivalent tree which is constructed using an algorithm which was originally developed for solving the multi-terminal maximum flow problem for undirected graphs.

While these early methods only deal with labels from a binary label set, most modern approaches that use graph cuts regularization have to solve problems with a label set consisting of multiple labels. In these cases the additional labels either lead to graphs with a very high complexity, or the graph cuts approach is applied iteratively by introducing moves in so called move spaces. Based on the work of Ferrari et al. [13] and Ferrari et al. [14] Veksler [48] and Boykov et al. [8, 9] show in their respective papers, methods for graph-based energy minimization that deal with that exact problem. Felzenszwalb and Zabih [12] also show different applications of graph techniques and dynamic programming for finding the solution to the energy minimization problem by using move spaces.

Since the graph cuts regularization approach is an important part of the method implemented for this master's thesis, it is given special attention in Section 4 that contains a theoretic definition of the graph cuts problem, illustrative examples, and a discussion of the solution for problems dealing with multiple labels by applying move making techniques.

2.5 Patch Prior

The inclusion of prior knowledge is a popular method to decide the likelihood of certain labelings. It is used to exclude impossible or improbable results. The most simple prior consists of the observation of neighboring pixels. The resulting probability is high for neighbors with the same label and low for different labels. Since these probabilities are used in the task of energy minimization, the pairwise terms describing the relationships be-

tween neighboring pixels take the form of a contrast sensitive Potts model and assign a low value for neighbors with identical values, and a value corresponding to the difference between the neighboring pixels otherwise [23].

Zoran and Weiss [53] show that priors that give high likelihood to the data also lead to good performance in patch restoration. They also demonstrate that patch based priors can be used to restore full images, and they introduce a new Gaussian Mixture prior which performs well on image denoising, deblurring, and inpainting.
In a first step they examine a number of popular prior models by comparing the log likelihood each model gives on a set of unseen natural image patches, and the performance of each of the models in patch denoising using MAP estimates. The results show that the higher the likelihood a model gives for a set of patches, the better it is in denoising them when they are corrupted. The goal is to find a reconstructed image in which every patch is likely under the obtained prior while keeping the reconstructed image as close to the corrupted image as possible by maximizing the Expected Patch Log Likelihood (EPLL) of the reconstructed image. This is achieved by an optimization step over all extracted noisy overlapping patches from the input image. The process of extracting patches and optimizing over all patches is iterated until convergence which is usually reached after four or five iterations.
The paper shows promising results for the task of image denoising. The framework improves the results of whole image restoration considerably when compared to simple patch averaging.

Kontschieder et al. [24] propose a simple and effective way to integrate structural information in random forests for the task of semantic segmentation. They show how random forests can be augmented with structured label information and introduce a new data splitting function that exploits the joint distributions observed in the structured label space for learning typical label transitions between object classes. The key concept of their work is that they take the structured labeling information of the label neighborhood into account. This information is included at the classification level which drastically improves the results and counteracts the assignment of meaningless configurations in the process.
A class prediction for a sample is obtained from a tree by recursively branching the sample down the tree until a leaf is reached. A class prediction for a sample is obtained from a forest by determining the majority of votes of the individual decision tree predictions. Each tree in the forest is trained independently on a random subset of the training set according to a recursive learning procedure. The standard random forest classifiers are in the pre-

sented approach extended by making them aware of the topological structure of the output label space.
In the case of this method the prior knowledge is learned by the propagation of the individual label patches in the decision trees of the random forests. The final labeling is then inferred from the individual decision tree predictions as the one receiving the majority of votes.

A more detailed discussion of patch priors and how the incorporation of patch prior knowledge is handled in the method developed for this master's thesis follows in Section 5.

This section presented related work on the topics of segmentation, with a special emphasis on semantic segmentation, conditional random fields, the graph cuts regularization approach, and patch priors.
The topic of semantic segmentation is again discussed in more detail in Section 3 where special attention is given to the interpretation of the semantic segmentation problem as an energy optimization problem. More details about the graph cuts approach are given in Section 4. Especially the solution for problems with a label set of more than two labels is examined thoroughly. Section 5 discusses the topic of patch priors in more detail and explains how the prior knowledge is incorporated in the method developed for this master's thesis.

3 State-of-the-Art Semantic Segmentation

The current state-of-the-art methods in semantic segmentation have in common that they use energy optimization techniques to solve the labeling problem. A general definition of the optimization problem is stated in Chapter 3.1. Chapter 3.2 presents the labeling problem in semantic segmentation as an energy optimization problem along with the design of the corresponding energy function, and Chapter 3.3 discusses the representation of the involved energies by so called potentials.

3.1 Optimization

The following definitions are derived from a combination of [12] and [48] and are identical or at least very similar for most of the related work on the topic. The solution to an optimization problem consists of selecting one possible solution f from a set of candidate solutions $\mathcal{S}$ and measuring its quality by the use of an objective function $E : \mathcal{S} \to \mathbb{R}$. The search space $\mathcal{S}$ for the candidate solutions has the size c^n, where c is the number of possible class labels and n denotes the number of individual sites that need to be labeled, which in the most basic case is the number of pixels in the image. So this search space usually comprises a very large set of possibilities, and the optimal solution to the optimization problem can theoretically be obtained by finding the global optimum of the objective function. The objective function E either measures the goodness or badness of a selected solution f and is therefore called energy maximization function or energy minimization function respectively. Since in the case of semantic segmentation we want to find a labeling that minimizes the energy, the problem is considered an energy minimization problem. The design of the resulting energy function is covered in detail in the next chapter. The optimal solution for the energy minimization problem is specified by the global minimum of the energy function

$$f^* = \arg\min_{f \in \mathcal{S}} E(f). \tag{1}$$

The problem of finding the global optimum of an arbitrary energy function is computationally intractable since the search space can consist of a very large number of candidate solutions, and to find the definitive optimum of a non-convex energy function, every single candidate needs to be considered. Yet, there exist approaches that examine a vast amount of the search space and can therefore be assumed to find a solution very close to the optimum. These methods include simulated annealing [21, 47] which was utilized for computer vision in [16], genetic algorithms [20], or the graph cuts approach that is covered in Section 4.

3.2 Energy Function

For computer vision problems, the energy function that is attempted to be optimized is always designed in the same fashion. In this model, image features are represented by a set of sites $\mathcal{P} = \{1, 2, .., n\}$. In the most basic case, sites represent single pixels, but they can also represent superpixels or other kinds of image patches or edges.
The labels to be assigned to the input image are specified by a set of labels $\mathcal{L} = \{l_1, l_2, .., l_k\}$, and the labeling problem consists of the task of assigning a label from label set $\mathcal{L}$ to each site in the set of sites $\mathcal{P}$. The labeling is a mapping from $\mathcal{P}$ to $\mathcal{L}$ and is denoted by $F = \{f_1, f_2, .., f_n\}$.
The energy of the labeling problem has the form

$$E(f) = E_{\text{Data}}(f) + \lambda E_{\text{Prior}}(f), \tag{2}$$

where the constant λ is a weighting factor that is a measure for the relative importance of the data and prior energy. It encodes whether the higher emphasis should be put on the data or the prior term, meaning which of the two kinds of energies should be considered as more reliable.
The data energy assigns large costs to labelings f which do not agree with the data and a small cost to labelings close to the data by

$$E_{\text{Data}}(f) = \sum_{p \in P} D_p(f_p). \tag{3}$$

where $D_p(f_p)$ measures how much assigning label f_p to pixel p disagrees with the data.
The prior energy in the same way assigns a heavy cost to the labelings f which are not likely from the point of view of the prior knowledge by

$$E_{\text{Prior}}(f) = \sum_{\{p,q\} \in \mathcal{N}} V_{\{p,q\}}(f_p, f_q). \tag{4}$$

The neighborhood system $\mathcal{N}$ is the set of all neighboring pairs $\{p, q\}$. The function $V_{\{p,q\}}(f_p, f_q)$ is called the neighbor interaction function, and penalizes neighboring pixels p and q if they have different labels.
The final general energy function has the form

$$E(f) = \sum_{p \in P} D_p(f_p) + \lambda \sum_{\{p,q\} \in \mathcal{N}} V_{\{p,q\}}(f_p, f_q). \tag{5}$$

The optimal solution for the labeling problem can be found by minimizing this energy function. One efficient way to achieve this goal is to apply graph cuts regularization which will be discussed in Section 4. The next chapter deals with the representation of the above energy function by the calculation of so called potentials on the input image.

3.3 Potentials

The data and prior energy from the above equations in the previous chapter can be represented efficiently by calculating so called unary and pairwise potentials respectively from the input image. The nature of these potentials and how they are obtained is described in detail in this chapter.

3.3.1 Unary Potentials

The unary potentials represent the data energy from Equation 3 and encode the constraints provided by the knowledge about the data. The goal is to get a representation of the input image that encodes the probability for a certain class from the label set f_p at every pixel location $D_p(f_p)$, where $f_p \in 1, .., c$ and c is the number of possible classes. The potentials are computed with a combination of color distribution, location, edge, and texture information from the input image by utilizing so called textons which were discussed in Chapter 2.2.1.
There are different methods to compute the unary potentials that are used in the further course of this work. One technique was presented by Shotton et al. with their paper on TextonBoost [42]. Another system is the Automatic Labelling Environment (ALE)[1] that was developed for [27]. The unary potentials used in this thesis were computed with the ALE. In the case of the MSRC-21 dataset there is also a second set of unary potentials available. These alternative potentials[2] are the same that were used in [25]. They are also used in this thesis to achieve the best comparison possible to this state-of-the-art method.
The actual potentials that are obtained this way consist of a 3-dimensional array of size height $\times$ width $\times$ number of classes where each value stands for the probability of a certain class at that pixel location.
Figure 8 shows an example image from the Corel-100 dataset with the corresponding groundtruth. The image at the right is the maximum likelihood estimation (MLE) that is derived from the unary potentials calculated with the ALE. The MLE estimates the most probable class at every pixel location based on the input consisting of the unary potentials.
The images in Figure 9 illustrate the nature of the unary potentials. Each semantic class in the dataset is depicted by an individual array of size height $\times$ width that indicates the most probable occurrences of that class in the input picture. Dark areas in the individual arrays denote that the probability of that class is low at the corresponding location. Light areas mean that

[1] http://cms.brookes.ac.uk/staff/PhilipTorr/ale.htm
[2] http://graphics.stanford.edu/projects/densecrf/unary/

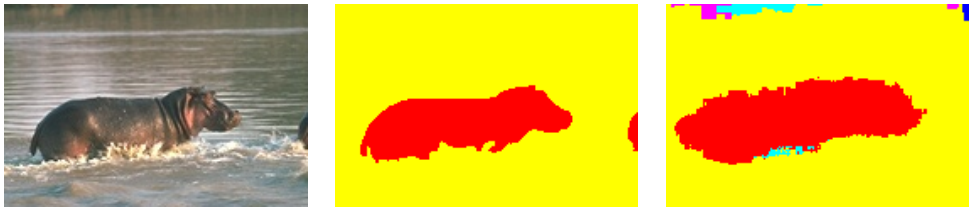

Figure 8: Example image from dataset Corel-100, corresponding groundtruth, and maximum likelihood estimation calculated on potentials computed with the ALE.

the probability of occurrences of the respective semantic class is high in that area. As can be seen, the two arrays that correspond to the semantic classes 'rhino/hippo' and 'water', that the input image solely consists of, have the brightest intensities, which means that they get the highest probability values throughout the seven possible semantic classes. The 'rhino/hippo' array (top, leftmost) has its highest values exactly at the hippo's location in the input picture, and the 'water' array (top, second from the right) has high values everywhere except at those areas where the hippo is located. The arrays corresponding to the classes 'polar bear' and 'sky' have the smallest intensity values which means that their probability is very low for this input picture. Another thing that can be observed is that the potentials are not perfect. Some of the arrays, denoting the other semantic classes that do not occur in the input image, have relatively high values in some areas as well. This can even lead to artifacts in the calculation of the maximum likelihood estimation, as can be seen in Figure 8 at the right, where some pixels are classified incorrectly. At the underside of the hippo, some pixels get labeled as belonging to the semantic class 'ground', and at the top of the image some classifications of 'ground', 'vegetation', and 'snow' do occur.

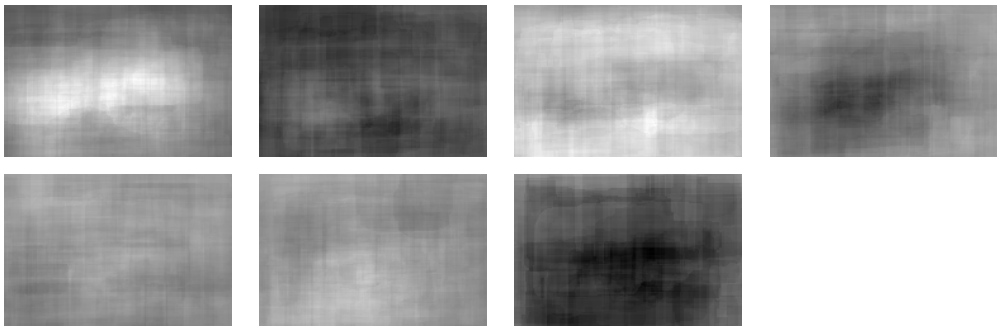

Figure 9: Unary potentials of an example image from dataset Corel-100 as computed with the ALE. The semantic classes are 'rhino/hippo', 'polar bear', 'water', 'snow', 'vegetation', 'ground', and 'sky' from top left to bottom right.

Unfortunately, the computation of the unary potentials is a very time consuming process. This is one reason why the developed system is not real-time capable. Fortunately, the unary potentials only have to be precomputed once and can then be stored for later use.

3.3.2 Pairwise Potentials

The pairwise potentials represent the prior energy from Equation 4 and encode the prior constraint. They reflect the compatibility of the local neighborhood of the pixels in the input image. Neighboring sites, in the most basic case individual pixels, are assigned a low value if they have similar image features, whereas sites that are next to each other and greatly differ in terms of image features are penalized with high values, according to the contrast sensitive Potts model by

$$\Phi_{p,q} = \begin{cases} 0, & \text{if } p = q \\ g(p,q), & \text{if } p \neq q \end{cases}, \tag{6}$$

where the function $g(p, q)$ assigns an energy according to the difference between sites p and q.
The difference in image features can mean different intensity values between neighboring pixels, or as in the case of the pairwise potentials that are used in the implementation of the semantic segmentation framework presented in this master's thesis, the assignment of identical or different labels.
The result is a sparse matrix of size (height $\times$ width) $\times$ (height $\times$ width) where each value stands for the relationship between the corresponding pixels. The matrix is sparse because it only holds the potential values for neighboring sites. These potentials are part of the input to the graph cuts framework which will be discussed in Section 4.

This section discussed the approach of considering the labeling problem in semantic segmentation as an energy optimization problem and presented the design of the energy functions. It also introduced the unary and pairwise potentials that are used to represent the energies that are involved in the process.
These potentials are the input for the graph cuts framework that is introduced in Section 4. Then the Sections 5 and 6 present the means that are taken to manipulate the unary potentials in order to include the learned patch prior knowledge to achieve the desired improvement of the segmentation results.

4 Graph Cuts

The task of solving energy minimization problems in computer vision can be reduced to the solving of a maximum flow problem in a graph. This task can in turn be efficiently solved with the graph cuts approach. The graphs used in this process are defined in Chapter 4.1. Chapter 4.2 illustrates the graph cuts approach with a simple example on a binary label set and Chapter 4.3 discusses the application of graph cuts regularization in the case of problems with more than two labels.

4.1 Definition

The weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ used in the graph cuts regularization is specified by a set of vertices $\mathcal{V}$ and a set of undirected edges $\mathcal{E}$. Each of the edges $e \in \mathcal{E}$ is assigned a non-negative weight w_e. The set of vertices includes two special nodes called source and sink, that are called terminals. Edges connecting two non-terminal vertices are called n-links. Edges that lead to one of the terminals are called t-links.
A cut $\mathcal{C} \subset \mathcal{E}$ is a set of edges that separates the terminals in $\mathcal{G}$: $\mathcal{G}(\mathcal{C}) = (\mathcal{V}, \mathcal{E} \setminus \mathcal{C})$. The cost of a cut equals the sum of the included edge weights: $|\mathcal{C}| = \sum_{e \in \mathcal{E}} w_e$. The goal is to find the cut with the smallest cost. This task is called the minimum cut problem and can be solved efficiently by determining the maximum flow between the terminals. The edge weights are in this case interpreted as capacity constraints [15].
Figure 10 shows a very simplified example of the application of the graph cuts approach for a binary label set of black and white pixels.

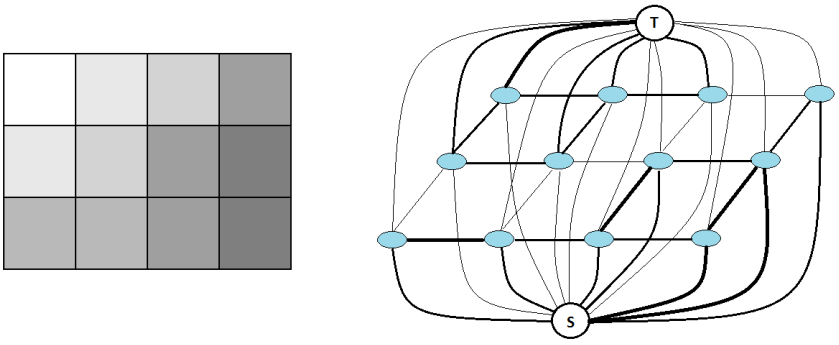

Figure 10: Example image (left) and corresponding graph with edge costs reflected by magnitude of edges (right).

The left side of Figure 10 shows a gray scale image with a size of 3×4. On the right the resulting graph is displayed with the twelve vertices corresponding to the twelve pixels of the image. The Terminals T and S denote the two possible label choices (in the case of this example white and black respectively). The t-links connect the terminals to the vertices and the n-links interconnect the non-terminal vertices. The magnitude of the edges corresponds to the respective edge costs. An example of a graph cut on the graph presented in Figure 10 and the resulting segmentation are shown in Figure 11. The cut that results from cutting the edges with minimal edge cost is denoted by the green line. The cut leaves each pixel vertex connected to only one terminal which leads to the segmentation presented on the right.

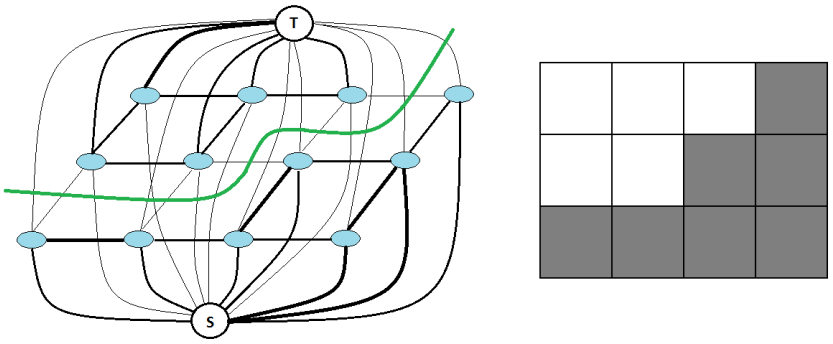

Figure 11: Example graph cut in the graph from Figure 10 (left) and resulting segmentation (right).

The next chapter demonstrates the theoretic definitions from above by means of a very basic example for a problem with a binary label set. After that some methods are presented that show how labeling problems with a larger label set can be solved.

4.2 Elementary Example

Chapter 3.3 in [12] contains an example for a problem with a binary label set, that illustrates the graph cuts approach very well by means of the restoration of a binary image:
The basis of the example is a binary image where each pixel has been corrupted independently. The goal is to clean up the input image which can be formulated as an optimization problem of the general energy function in Equation 5 from the previous chapter. The search space is $\{0,1\}^n$ where n is the number of pixels in the image which leads to 2^n possible solutions.

A unary function D_p corresponding to the data energy from Equation 3 provides the cost for labeling the p-th pixel using one of the two possible values. The cost is zero for pixels that have the same label that was observed in the corrupted image and a positive value λ if the pixel has the opposite label.
A binary Function $V_{p,q}(x_p, x_q)$ corresponds to the prior energy in Equation 4 and is 1 if the two adjacent pixels x_p and x_q have different labels and 0 otherwise.
The resulting energy function that is similar to Equation 5 leads to a value of λ times the number of pixels that get assigned a different label from the one observed in the corrupted image plus the number of adjacent pixels with different labels. Minimization of this energy leads to a spatially coherent labeling that is similar to the observed data.
The resulting graph consists of an array of vertices corresponding to the array of pixels in the image. The vertices are connected corresponding to their neighboring pixels in the image. Each pixel vertex is connected to the two terminals which correspond to the labels 0 and 1 which is similar to the example in Figures 10 and 11.
A cut in this graph leaves each pixel vertex connected to exactly one terminal vertex which complies to a binary labeling. The cost of the cut is the energy that is associated with a certain labeling. A cut between a pixel vertex and a terminal corresponds to the function D_p and the function $V_{p,q}$ is responsible for a cut between two adjacent pixel vertices.
This simple example shows the graphs cuts approach for a binary label set. In many cases the labeling problem in computer vision consists of label sets of higher magnitude. The next chapter presents methods to deal with those cases.

4.3 Multiple Labels (Move Spaces)

The construction of the graph used in the elementary example above can be generalized to accommodate for multiple labels without introducing any further terminals.
One possibility is to build a graph with additional edges per pixel. The simplest solution is to construct a chain of $k-1$ vertices per pixel which leads to k possible edges that may be cut (one per label). This approach is limited to functions $V_{p,q}$ that are convex. Additionally it leads to graphs with $n \times k$ nodes which for datasets with a lot of different labels can lead to memory complexity problems.
A second approach is to iteratively apply the graph cuts approach. This is achieved by introducing the notion of a move from an initial labeling where each pixel can either keep its current label or adopt a new one. In this way

each pixel is again faced with a binary decision similar to the one stated in the example above.
There are different types of moves to alter an initial labeling, and the possible moves are discerned by their corresponding move spaces. The following definitions are from [48]:
In general a move space $\mathcal{M}$ is specified by a set of moves $\mathcal{M} \subset \mathcal{F} \times \mathcal{F}$. A move is a pair of labelings $(f, f') \in \mathcal{F} \times \mathcal{F}$, where $\mathcal{F}$ is the set of all possible labelings. If $(f, f') \in \mathcal{M}$ then f is one move away from f'. A labeling f is considered a local minimum with respect to the move space $\mathcal{M}$ if $E(f) \leq E(f')$ for any $(f, f') \in \mathcal{M}$. A local minimum is defined by the moves that are allowed. If $\mathcal{M} = \mathcal{F} \times \mathcal{F}$ then the local minimum with respect to $\mathcal{M}$ is also the global minimum. For every move space there exist $O(2^n)$ possible moves for any labeling f.
There are four different types of move spaces that are discerned by the labeling action that is performed on the initial labeling f, which in turn lead to the new labeling f':
In the **relabel move space** moves are indexed by a pair of labels $\{\alpha, \beta\} \subset \mathcal{L}$. When an α-β relabeling move is performed some pixels that previously had the label α are relabeled with label β.
The **swap move space** is again defined by a pair of labels $\{\alpha, \beta\} \subset \mathcal{L}$. During an α-β swap move pixels with label α switch labels with pixels that are labeled with label β.
In the **jump move space** moves are defined by a one-to-one function $h : \mathcal{L} \rightarrow \{0, 1, .., k-1\}$, where k is the number of labels. This function is needed to assign each label an integer value if necessary. Jumps are labeled by an integer $i \in \{0, 1, .., k-1\}$ and called i-jumps. When an i-jump is performed the labels of some pixels are changed by the value i. The relabel move space is contained in the jump move space.
Finally, the **expansion move space** is defined by a single label $\alpha \in \mathcal{L}$. When an α-expansion move is performed a set of pixels in labeling f switches the current label to label α to derive the new labeling f'.
As discussed, most of the state-of-the-art work on the topic of semantic segmentation uses the approach of energy minimization with graph cuts. Since most of the segmentation problems deal with more than two labels, the above techniques of using move spaces are applied in most cases where the graph cuts approach is used.
The graph cuts framework that is used in this thesis is viewed as a black box that takes as input the unary and pairwise potentials as well as parameters that determine the relative importance of the potentials, and delivers a semantic segmentation as output. The framework also offers the choice of either using the swap or expansion move space. For all the experiments that

were performed and that are documented in Section 9 the swap move space was chosen.

This section presented the graph cuts approach by giving a theoretic definition of the graph and illustrating it with simple examples for the case of a binary label set. It also discussed methods for the iterative application of the graph cuts regularization with the help of moves in different move spaces. The next section shows how the patch prior knowledge can be incorporated to manipulate the unary potentials discussed in Chapter 3.3.1 that are then used as input for the graph cuts framework.

5 Patch Prior

The goal of this thesis is to incorporate prior knowledge on the semantic level to obtain a sound semantic segmentation. This section presents the idea the implemented method is based on in Chapter 5.1. Chapter 5.2 gives an explanation of the motivation behind the use of prior knowledge on the patch level for semantic segmentation. Finally, some details on the generation of the patch prior database itself are given in Chapter 5.3.

5.1 Basic Idea

The basic idea of the method presented in this master's thesis is inspired by the approach of Zoran & Weiss [53], where the concept of learning patch priors is used for the task of image denoising. The goal is to find a reconstructed image in which every patch is likely under the obtained prior, while keeping the reconstructed image as close to the corrupted image as possible. The expected patch log likelihood of the reconstructed image is maximized to meet the first constraint, while as a second constraint the similarity between the reconstructed image and the corrupted image should be as close as possible.
This method is used as basis for the system developed for this master's thesis. Similar to the first constraint in [53] every single patch of the input image should have a high likelihood under the learned prior, while as a second constraint the whole image should be segmented in a semantically sound way. This second constraint is met by averaging on the semantic level by applying the graph cuts approach that was presented in Section 4.

5.2 Motivation

The goal of incorporating prior knowledge is to learn which label configurations are common and which are not. In this way it becomes possible to rule out labelings that semantically do not make any sense and assign a high likelihood to labelings that occur frequently in the learned patch prior database. For example a learned prior might state that pixels labeled with the semantic class label 'sky' will always be at the top part of patches whereas pixels labeled with the label 'road' will be at the bottom of the image patch most of the time. When this learned prior knowledge is applied to patches of an input image a patch with a configuration of 'road' pixels at the bottom and 'sky' pixels above will have a high confidence whereas a patch with 'sky' pixels at the bottom and 'road' pixels on top will be voted as being highly improbable.

5.3 Implementation

The prior knowledge is learned by analyzing a set of training data which consists of the annotated groundtruth of training images on the semantic level. For each pixel in every training image a label patch of a certain quadratic size is extracted with the pixel as the center. All label patches are extracted from all training images, and a histogram of their occurrences is built. The number of occurrences of each label patch in the prior database, that is obtained this way, is a measure for the probability that it might also occur in an input image for the semantic segmentation. It also gives a first impression of label configurations that are common and configurations that are impossible.
The design of the actual patch prior database was one of the major challenges of the practical part of this thesis. The requirement was that access to the database could not be too costly in terms of time since there would be a lot of database queries for the individual patches. There are on the one hand a lot of accesses during the generating of the database, where new patches have to be entered in the database if necessary, or values for patches that are already present have to be updated. On the other hand, during the two-stage iterative process of updating the unary potentials and applying the graph cuts framework, which will be discussed in the next section, there are a lot of queries to update the probabilities for the patches of the intermediate results. Since the design of this database was so crucial in the process of developing the system, Section 8 is dedicated to the description of the database structure and a detailed analysis of the conclusions that can be drawn from the learned patch prior knowledge.
Like the precomputation of the unary potentials, as stated in Chapter 3.3.1, the precomputation of the patch prior database can be a very time consuming effort, especially for datasets with big images, a large amount of training data, and in the case of the larger patch sizes. This is the second reason why the developed system is not real-time capable. Fortunately, it is again only necessary to compute the patch prior database once for a certain patch size and a certain dataset, and the database can then be stored for later use.

This section presented details of the patch prior knowledge that is in the course of this master's thesis used to avoid impossible or improbable label configurations. It introduced the idea that the presented method is based on and the motivation behind the use of prior knowledge. It also gave a short overview of the details of the implementation of the patch prior database.
The next section shows how the precomputed patch prior database, along with the graph cuts approach is utilized in the presented semantic segmentation framework.

6 Method

This section discusses the semantic segmentation framework that was implemented in the course of this master's thesis. Chapter 6.1 presents a short overview of the system, with pre-existing parts and parts that needed to be implemented. Chapter 6.2 presents the core of the implemented method with the two-stage iterative process that merges the previously presented approaches of semantic segmentation considered as an energy optimization problem as introduced in Section 3, the graph cuts approach with the unary and pairwise potentials as presented in Section 4, and the incorporation of the patch prior knowledge which was discussed in Section 5, to yield the final results. This section is concluded by a step-by-step example that illustrates the whole process in Chapter 6.3.

6.1 Overview

The goal of this thesis was to build a robust semantic segmentation framework by introducing prior knowledge on the semantic level, that is learned on the patch level, for the inference method. Figure 12 shows an overview of the presented semantic segmentation system with its input, individual components, intermediate results, and output. For the gray components already existing components could be used. The blue components had to be implemented for this master's thesis. The resulting system is then tested on the databases described in Section 7, and the resulting segmentation results are compared to those achieved by the state-of-the-art methods mentioned in Chapter 2.2.

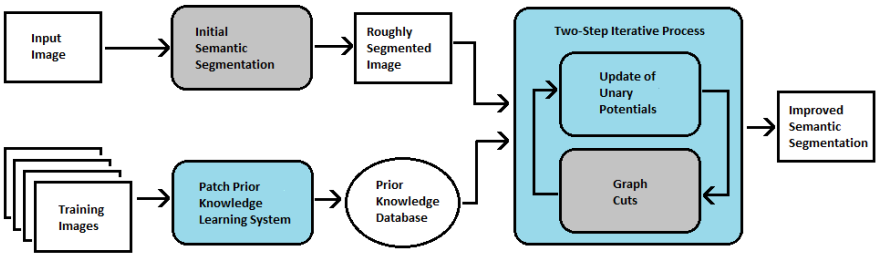

Figure 12: Overview of the semantic segmentation framework: gray components already existed, blue components had to be implemented.

The input for the semantic segmentation system consists of two parts. The first part is a rough semantic segmentation in the form of the unary potentials as discussed in Chapter 3.3. The second part of the input consists of the previously learned patch prior database. The initial segmentation and the patch prior database are used in the core of the system, a two-stage iterative process, which desirably leads to a gradual improvement of the initial segmentation. The details of this iterative process, consisting of an alternating application of an update of the unary potentials and the graph cuts regularization, are discussed in the next chapter.

6.2 Two-Stage Iterative Process

This chapter presents the heart of the semantic segmentation framework that was implemented for this master's thesis. It combines the approaches discussed in the previous sections by incorporating the learned prior knowledge with the utilization of the previously built patch prior databases. The initial segmentation represented by the precomputed unary potentials is iteratively updated and graph cuts regularization is applied until convergence. The following chapters discuss the details of this iterative process.

6.2.1 Incorporation of Patch Prior Knowledge

Every pixel of the input image is assigned a probability for each of the possible semantic categories according to the prior knowledge that was learned during the learning phase when the patch prior databases were built. All this happens at the semantic level which means that the input image contains a label at each pixel location instead of an intensity value. A label patch of a certain quadratic size is extracted around the pixel that is currently under examination. The pixel, that is automatically the center pixel of the extracted label patch, is varied over all c possible classes. The c resulting label patches, that are obtained in this way, are used as query patch for the patch prior database and in that way their respective number of occurrences is determined. The individual values are stored in a vector of length c which is then normalized to sum up to one. The corresponding values reflect the probability for the pixel as belonging to each of the c possible classes.
The following example illustrates the approach of the incorporation of the prior knowledge. An example label patch of size 3×3 is extracted from the input image. The label patch has the following label configuration:

2	2	3
2	**3**	3
3	3	3

The center pixel is varied over all four possible label classes and the resulting label patches are used as query in the previously learned patch prior database which returns the following numbers of occurrences for the respective patches:

2	2	3
2	**1**	3
3	3	3

: 0

2	2	3
2	**2**	3
3	3	3

: 10

2	2	3
2	**3**	3
3	3	3

: 40

2	2	3
2	**4**	3
3	3	3

: 0

The resulting vector of occurrences is $[0, 10, 40, 0]$ and after normalization $[0.0, 0.2, 0.8, 0.0]$ which corresponds to a probability of 20% for the pixel under examination being of class 2, and a much higher probability of 80% that the pixel is actually of class 3.
This process is repeated for every pixel of the input image and the result is an update of the unary potentials under the previously learned prior. The details of this update are discussed in the next chapter.

6.2.2 Update of Unary Potentials

The incorporation of the prior knowledge discussed in the previous chapter is the basis for the update of the unary potentials in the two-step iterative process. What happens is that Equation 5 gets extended to

$$E(f) = \gamma \sum_{p \in P} D_p(f_p) + \lambda \sum_{\{p,q\} \in \mathcal{N}} V_{\{p,q\}}(f_p, f_q) + (1 - \gamma) \sum_{p \in P} \widetilde{D}_p(f_p), \qquad (7)$$

where the function $\widetilde{D}_p(f_p)$ stands for the energy introduced by the patch prior knowledge and the variable γ controls the relative importance of the original prior term and the newly introduced patch prior term respectively. Since the update of the unary potentials uses the result of the previous iteration only as basis and does not take it into further account, γ is chosen with a value of 0 in the presented implementation.
For the first update of the unary potentials the basis for the incorporation of the patch prior knowledge is the maximum likelihood estimation (MLE) on the actual precomputed unary potentials which yields an array with the most likely of the possible class labels at each pixel location. The prior is included as discussed in the previous chapter and the resulting array of probabilities replaces the previous unary potentials. These unary potentials are then used as part of the input of the graph cuts framework, as presented in the next chapter. The result of the graph cuts regularization is then used as basis for all further iterations, and the patch prior knowledge is again used as described in Chapter 6.2.1 to again yield an updated version of the unary potentials.

6.2.3 Graph Cuts

The updated unary potentials and the pairwise potentials, which are calculated from the input image and stay the same throughout the whole process, are used as input for the graph cuts framework. The input also includes two parameters α and β that control the relative importance of the unary and pairwise potentials respectively. A high value for α and a low value for β means that the graph cuts framework should lay great importance on the unary potentials which is equivalent to putting more trust in this data and a lower trust in the other. Since the focus of this thesis lies on the inclusion of patch prior knowledge, which is achieved by the manipulation of the unary potentials, the system was tested with values for these parameters that always laid a high importance on the unary potentials and only a low importance on the pairwise potentials.
The output is a new semantic segmentation that is achieved by applying graph cuts regularization by utilizing the swap move space. The resulting labeling is the basis for the next iteration.

6.2.4 Iteration

The above steps of updating the unary potentials and applying the graph cuts approach is repeated iteratively. After every iteration the new segmentation is compared to the old one (in case of the second iteration with the MLE on the unary potentials) and the pixelwise difference is calculated. This process is repeated until one of the following stop criteria is met:

- Maximum number of iterations is reached.
- Difference between segmentation results is smaller than threshold.

The first criterion simply exists to prevent the system from getting caught in an infinite loop. The method usually converges very fast, and the maximum number of iterations is set to the low value of 10. For the second criterion a threshold that depends on the size of the image to be segmented and the chosen patch size is calculated. Whenever the difference between a new segmentation result and the previous segmentation is smaller than or equal to this threshold, the process is stopped with the output of the graph cuts framework as the final result. Theoretically, especially if the threshold for the second criterion is chosen too low, it could happen that the system reaches a state where the segmentation results oscillate between two very similar results, but their difference is higher than the chosen threshold. In that case the first criterion stops the execution. Usually this is not necessary and the method converges before the maximum number of iterations is reached.

6.3 Step-by-step Example

This section presents a step-by-step example of the semantic segmentation process performed on a single image by the system developed in the course of this master's thesis. The steps are executed in the sequence described in the previous chapter and the intermediate results are given to illustrate the process.
The sample image from the Corel-100 dataset that is used to exemplify the method is the same image that was used in Chapter 3.3.1 to visualize the unary potentials. The unary potentials were calculated with the Automatic Labelling Environment and the parameters for the weight of the unary and pairwise potentials respectively are $\alpha = 500$ and $\beta = 10$ which lays a heavy importance on the unary potentials. The patch size chosen for this example is 11×11.
The input for the first iteration consists of the unary potentials calculated for the image on which the maximum likelihood estimation is performed. The potentials can be viewed in Figure 9 and the corresponding MLE is shown on the right in Figure 8 which can both be found in Chapter 3.3.1. This MLE is then used as the basis for the update of the unary potentials by applying the learned patch prior knowledge from the corresponding database. All unique patches are extracted from the MLE and the probabilities of the patch variations are determined by querying the database. The resulting new unary potentials can be viewed in Figure 13 where only those arrays are shown that contain any information. The images from left to right correspond to 'rhino/hippo', 'water', 'vegetation', and 'ground'.

Figure 13: Iteration 1: Updated unary potentials for the classes 'rhino/hippo', 'water', 'vegetation', and 'ground' (from left to right).

As can be seen the main information is justifiably reduced to the two semantic classes 'rhino/hippo' and 'water'. The arrays corresponding to the classes 'vegetation' and 'ground' hardly contain any information at all and the remaining three classes are completely empty.
The unary potentials are used as input for the graph cuts framework along with the pairwise potentials calculated from the input image and the parameters α and β. The result of the graph cuts regularization can be viewed in Figure 14.

Figure 14: Iteration 1: Segmentation before (left) and after (middle) graph cuts regularization and difference between segmentations (1537 pixels).

As will be seen in the course of this example most of the change takes place during the first iteration. The small artifacts of the wrong class labels 'vegetation' and 'ground' are obliterated and the shape of the hippo gets smoothed at the edges. This can be seen in the segmentation results as well as in the images for the unary potentials. For the semantic class 'water' the previously wrongly classified areas at the top are now correctly classified as water. The pixelwise difference between the two segmentations amounts to 1537 pixels. In the next iteration the actual output of the graph cuts framework is the basis for the update of the unary potentials. Figure 15 shows only the updated unary potentials for the semantic classes 'rhino/hippo' and 'water', since none of the other arrays contain any more information. The previous segmentation and the new segmentation are also shown along with the difference which is 286 pixels for this iteration.

Figure 15: Iteration 2: Updated unary potentials for semantic classes 'rhino/hippo' and 'water', segmentation before and after graph cuts regularization, and difference between segmentations (286 pixels) (left to right).

There are still some obvious changes between the two segmentations during this iteration. The most profound one is the smoothing of the frayed area at the head of the hippo. Figure 16 shows the results of the next iteration.

Figure 16: Iteration 3: Updated unary potentials for semantic classes 'rhino/hippo' and 'water', segmentation before and after graph cuts regularization, and difference between segmentations (24 pixels) (left to right).

There is again some minor change in the updated unary potentials which results in a slight change of the segmentations. Some pixels labeled as 'rhino/hippo' in the previous segmentation change to the class label 'water' at the bottom left of the hippo. The difference is only 24 pixels now. Figure 17 shows the outcome of the final iteration.

Figure 17: Iteration 4: Updated unary potentials for semantic classes 'rhino/hippo' and 'water', segmentation before and after graph cuts regularization, and difference between segmentations (0 pixels) (left to right).

Again, there are some small changes between the corresponding unary potentials for the classes 'rhino/hippo' and 'water' respectively but they apparently have no influence on the output of the graph cuts regularization. The new segmentation is identical to the previous one. The difference is 0 pixels which triggers the system to abort the iterative process and to consider the last output of the graph cuts framework to be the final segmentation result. Figure 18 shows once again the original image and the corresponding groundtruth along with the final segmentation result for comparison.

Figure 18: Original image, corresponding groundtruth, and final semantic segmentation result after 4 iterations.

Admittedly, the example image chosen for the above demonstration is one that yields particularly good results. The final segmentation is very close to the optimal segmentation as represented by the annotated groundtruth. The segmentation accuracy in this case for the classes 'rhino/hippo' and 'water' amounts to 94.66% and 96.86% respectively.

This section presented the details of the semantic segmentation method that was used in the course of this master's thesis. It discussed the inclusion of the previously learned patch prior knowledge and the two-stage iterative process that includes the replacement of the unary potentials by newly constructed

ones that are more probable under the learned prior, and a graph cuts regularization step that uses these newly constructed potentials as input. The section was concluded by the simulation of a complete run of the system which illustrates the details of each stage in the application of the framework by describing each step and the intermediate results.
The next section shortly introduces the different datasets that were used in the experiments and evaluation of the developed method. Then the details of the construction and querying of the actual patch prior databases are discussed in Section 8. Section 9 presents the results that could be achieved with the method presented in this section.

7 Databases

This section introduces the databases that are used in the evaluation process which follows in Section 9. Each dataset is presented with a short description of the images, their size and number and the split into training and test data. Each of the datasets includes a manually annotated groundtruth of the semantic class labels it contains. The segmentation results achieved by some of the methods mentioned in Chapter 2.2 are reported as well.

7.1 Corel-100

The Corel-100 dataset is a subset of 100 images of the Corel Image Database[3] and consists of images of African and Arctic wildlife. The images are each of size 180×120 pixels and contain photographs of rhinos, hippos, and polar bears. The images were manually labeled into seven classes which are 'ground', 'polar bear', 'rhino/hippo', 'sky', 'snow', 'vegetation', and 'water'. The dataset contains with the Arctic and African wildlife scenes only two types of different images and only comparatively few semantic classes which makes it particularly easy to work with. This dataset also has the advantage that 100% of the pixels in the images are labeled as belonging to one of the seven classes.
Figure 19 shows example images and the corresponding groundtruth from the subset of the Corel Image Database.

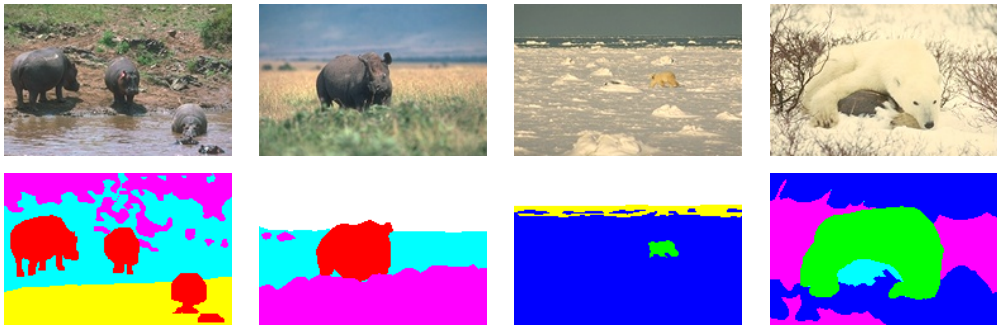

Figure 19: Example images and corresponding groundtruth for the Corel-100 dataset.

The Corel-100 dataset is used in [19] where an average classification rate of 80.0% can be achieved, in [42] where 74.6% can be reached, and in [46] that could achieve average labeling accuracies of 83.0%.

[3] http://www.cs.toronto.edu/~hexm/label.htm

Table 3: Achieved accuracies with different methods on dataset Corel-100.

Method	**accuracy**
Shotton et al. [42]	74.6%
He et al. [19]	80.0%
Toyoda & Hasegawa [46]	83.0%

Since no information about the division into training and test data was available for these methods, a randomized division into 50% training images and 50% test images was chosen, which led to 50 images in each category.

7.2 MSRC-21

The Microsoft Research Cambridge Object Recognition Image Database[4] [41] contains 591 color images of size 320×213 including 23 semantic classes only 21 of which are typically used. The 21 classes are 'aeroplane', 'bicycle', 'bird', 'boat', 'body', 'book', 'building', 'car', 'cat', 'chair', 'cow', 'dog', 'face', 'flower', 'grass', 'road', 'sheep', 'sign', 'sky', 'tree', and 'water'. The classes 'horse' and 'mountain' are ignored.
Figure 20 shows examples of images from the MSRC-21 database and the two sets of corresponding annotated groundtruth available.

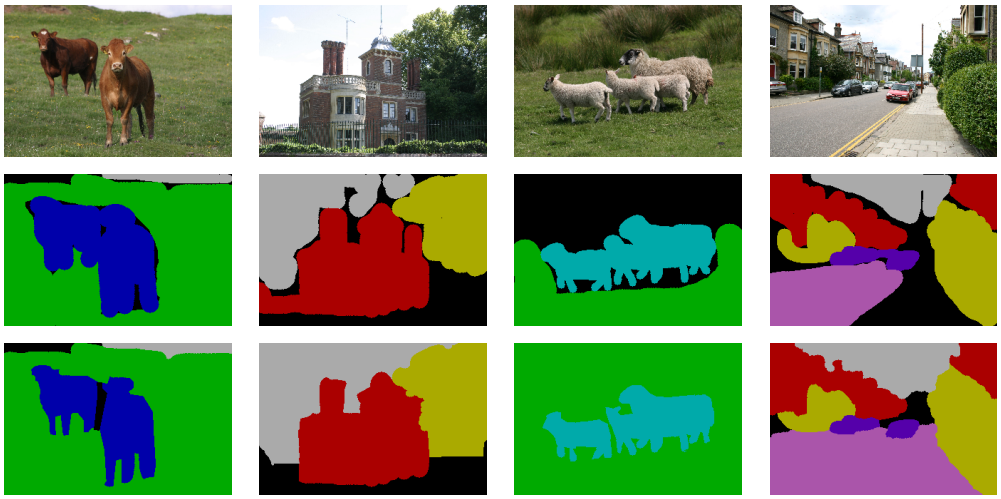

Figure 20: Example images (top), corresponding groundtruth as used in [25] (middle), and relabeled groundtruth (bottom) for the MSRC-21 dataset.

[4]http://research.microsoft.com/en-us/projects/objectclassrecognition/

There are two different sets of groundtruth data available for the MSRC-21 dataset. One is the set that is also used in [25] for which the labeling of the groundtruth images is very imprecise. Especially at the boundary between segments of different classes, in many cases a lot of pixels are classified as 'void'. This leads to problems with the presented method since patches with the 'void' class will be learned, but the classifier will not classify pixels as 'void' in the first segmentation step that yields the rough initial segmentation. The second set of groundtruth data is a relabeled version of the first one in which many of the pixels previously declared as 'void' are labeled as belonging to one of the 21 classes in the dataset. As the examples in Figure 20 show, there are still many areas that are left unlabeled.
The MSRC-21 dataset has the advantage that precomputed unary potentials are available that are used in [25]. It is therefore possible to compare the implemented method to this state-of-the-art method and also to analyze the impact of different kinds of unary potentials on the segmentation results.
The MSRC-21 dataset is used in [25] where an average segmentation accuracy of 86.0% (global) and 78.3% (average) can be achieved on the standard groundtruth. The dataset is also used in [42] where an average accuracy of 72.2% is reached and in [27] where an accuracy of 87% (global) and 77% (average) can be achieved.

Table 4: Achieved accuracies with different methods on dataset MSRC-21.

Method	**accuracy**	
	global	**average**
Shotton et al. [42]	-	72.2%
Krähenbühl & Koltun [25]	86.0%	78.3%
Ladický et al. [27]	87%	77%

The division into training and test data that is used in [25] splits the images into 276 training images, 256 test images, and the rest of 59 images for validation. This split is also used in this master's thesis to achieve the best possible comparison to [25]. As for all the other datasets the potentials were also calculated with the Automatic Labelling Environment (ALE)[5]. In the course of these computations the data was randomly split into 50% training and 50% test images. To examine the influence that different sets of potentials can have on the segmentation results, a new test set was built from the intersection of the two test sets, resulting in a new test dataset of 138 images.

[5]http://cms.brookes.ac.uk/staff/PhilipTorr/ale.htm

7.3 Sowerby Image Database of British Aerospace

The Sowerby Image Database of British Aerospace[6] [11] features a set of color images of outdoor scenes and the associated labels. The 104 images contain objects near roads in rural and suburban areas and include seven label classes which are 'building', 'car', 'road marking', 'road surface', 'sky', 'street object', and 'vegetation'. The images included in this dataset are with a size of only 96×64 pixels the smallest featured in any of the datasets. This fact leads to the problem that the images include very fine structures like the road markings in the middle of a road that are only very few pixels wide which leads to problems with the implemented method as will be shown in Section 9.
Figure 21 shows example images and the corresponding groundtruth from the Sowerby Image Database of British Aerospace.

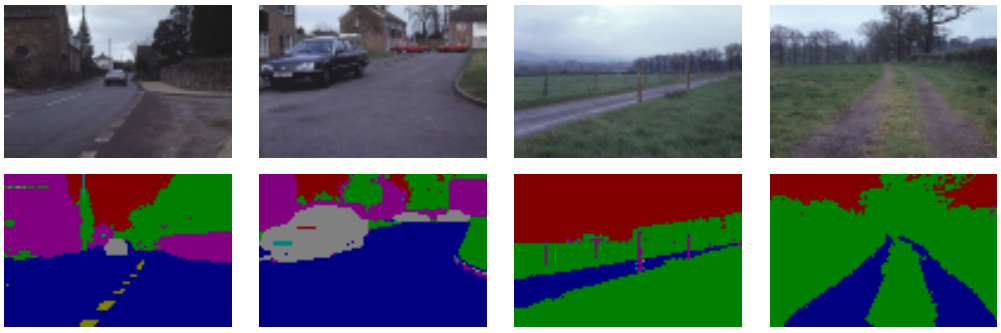

Figure 21: Example images and corresponding groundtruth for the Sowerby dataset.

The Sowerby dataset is used in [19] where an average classification rate of 89.5% can be reached, in [42] where an accuracy of 88.6% can be achieved, and in [46] with an average labeling accuracy of 90.0%.

Table 5: Achieved accuracies with different methods on dataset Sowerby.

Method	**accuracy**
Shotton et al. [42]	88.6%
He et al. [19]	89.5%
Toyoda & Hasegawa [46]	90.0%

The Sowerby dataset was again randomly split into 50% training images and 50% test images which results in 52 images in each of the two categories.

[6] http://www.robots.ox.ac.uk/~lubor/

8 Patch Prior Database Design and Analysis

This section presents a detailed discussion of the patch prior database that was implemented for the learning and application of the prior knowledge. Chapter 8.1 discusses the structure of the database including the generation and query process. Chapter 8.2 deals with the influence of the patch size on the quantity of extracted patches as well as the nature of these patches, and presents the results for the different datasets in the form of tables and diagrams.

8.1 Patch Prior Database Design

As already mentioned in Chapter 5.3, the actual design of the patch prior database used in the system implemented for this master's thesis was the most crucial step. Since the database has to be accessed multiple times during the generation as well as in the process of the actual application, the implementation had to comply with the requirement that the patch prior knowledge, that was extracted from the training data, could be stored efficiently. The precomputed data should not require a large amount of memory on the one hand, and also be accessible in a very efficient manner on the other hand.
All that needs to be stored in the database is the number of occurrences for each of the unique patches. The easiest way to achieve this goal would be some kind of hash function that assigns a unique hash value to each individual patch. In this way the resulting unique numerical value could then be used as index in a simple one-dimensional vector that only holds the number of occurrences for the corresponding patch at the index location that is defined by the calculated hash value.
Unfortunately, the design of such a hash function is not trivial or might even be completely impossible for an arbitrary number of classes in the dataset and arbitrary patch sizes. Since the use of a unique hash function for each label patch was infeasible, the approach was relaxed to an approximation. Instead of requiring a hash function to only yield one unique hash value for each individual patch, the property was relaxed to a hash function that yields identical hash values for only a very small number of different patches. In that way it is possible to store the patches and the number of their respective occurrences consecutively in the sequence of their incidence, which is more memory-efficient than updating an empty database with one entry for every possible patch. The hash values are utilized to build the first step of an index system, which allows access to the individual database locations in an almost random access fashion.

The following chapter discusses the method of calculating the hash value for the image patches. Then the process of querying the database and the generation of the database from the training data are discussed. Examples are given for each of the processes for illustration purposes.

8.1.1 Hash Function

The hash function that was discussed in the introduction to this section consists of simple matrix operations that can be performed very efficiently with MATLAB®. The label patches used at this point consist of an array of size $k \times k$, where k is the patch size. The array holds numerical values that stand for the corresponding label classes.
The following examples represent label patches of size 3×3. The five label patches presented, correspond to those extracted in the example in Figure 4 in Section 1.

2	2	2
2	**2**	2
2	2	2

2	2	2
3	**3**	3
3	3	3

2	2	3
2	**3**	3
3	3	3

3	3	2
3	**3**	2
3	3	2

3	3	3
3	**3**	3
3	3	3

The example label patches consist of the class labels 'grass' (2) and 'cow' (3). The center pixels, marked in bold, are used in a later stage of the indexing process.
Each label patch is multiplied element-wise with a mask matrix and the sum over all rows and columns of the result is calculated to obtain a single integer value. This calculation sequence is equivalent to the following two steps. First the patch is reshaped into a vector:

$$\begin{array}{|c|c|c|c|c|} \hline p_{1,1} & p_{1,2} & .. & p_{1,k-1} & p_{1,k} \\ \hline p_{2,1} & p_{2,2} & .. & p_{2,k-1} & p_{2,k} \\ \hline .. & .. & .. & .. & .. \\ \hline p_{k-1,1} & p_{k-1,2} & .. & p_{k-1,k-1} & p_{k-1,k} \\ \hline p_{k,1} & p_{k,2} & .. & p_{k,k-1} & p_{k,k} \\ \hline \end{array} \rightarrow \overbrace{\begin{bmatrix} p_{1,1} & p_{2,1} & .. & p_{k-1,k} & p_{k,k} \end{bmatrix}}^{k^2 \text{ elements}}$$

Then the scalar product of the resulting vector with a mask vector is computed by multiplying the vectors element-wise. The mask vector has to stay the same for the generation and for the querying of the database. The mask that was chosen for this purpose is simply a vector that contains the numbers from 1 to k^2 where k is the patch size:

$$\begin{bmatrix} p_{1,1} & p_{2,1} & .. & p_{k-1,k} & p_{k,k} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \\ .. \\ k^2-1 \\ k^2 \end{bmatrix} = \text{hash value}$$

In this way a scalar hash value can be calculated for each patch. As mentioned before, this value is not unique, but it has the property that only very few different patches hash to the same value. For example, both of the following patches of size 3×3:

1	1	1
1	**1**	1
2	1	1

and

2	1	1
2	**1**	1
1	1	1

hash to the same hash value of 48 since $1\times1+1\times2+2\times3 = 2\times1+2\times2+1\times3$. The hash values calculated in this manner are then used as the first stage of an indexing system, that is utilized to allow efficient access to the individual database locations. The process of querying the database is presented in the next chapter.

8.1.2 Database Query

As a first step to reduce the search complexity, the database is subdivided into a number of sub-databases, one for each semantic class that occurs in the dataset. The center pixel, marked in bold in all examples of label patches, is used as an index to the corresponding sub-database. This fact is illustrated in Figure 22.

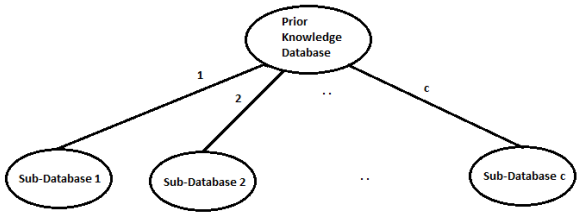

Figure 22: Database with sub-databases for dataset with c semantic classes.

Each of these sub-databases can then be queried separately. The sub-database is chosen according to the center pixel of the image patch that is searched for. The next indexing stage consists of calculating the hash value according to the previous chapter. This hash value is used as index to a data structure

for all label patches that hash to this same value which holds their respective locations in the next stage of the query.
This process is illustrated with an example of an excerpt from an actual database built for the MSRC-21 [41] dataset, which was presented in Chapter 7.2, for a patch size of 3×3. The exemplarily presented sub-database is the one for patches with center pixels of label class 1:

hash index	1	2	..	44	45	46	47	48	49	50
location(s)	{ }	{ }	..	{ }	1	454	95	455 456	283 457 458	36 459 460

The first hash index that points to a part of the data structure that holds any actual information has the value of 45 and holds the location in the next query stage of the homogenous patch that only holds pixels from label class 1 because $1 \times 1 + 1 \times 2 + .. + 1 \times 9 = 45$. The hash index at hash index 48 is the first that holds the location information for more than one patch.
The locations that are retrieved in this way by the hash value index are examined consecutively. The data structure at this next stage of the query holds the composition of the actual label patch and the number of its occurrences in the training data. The query label patch is compared to the label patch at the locations that were determined with the hash index, and as soon as the patch at the specified location matches the query patch, the number of its occurrences is returned as the query result.
The example from the previous paragraph is continued by accessing a portion of the next indexing stage, that shows the two patches that hash to the value 48 from the previous example at locations 455 and 456 with their respective number of occurrences of 35 and 4 respectively. It also shows the two patches at locations 457 and 458 that hash to the identical hash value of 49. The example excerpt also shows that similar image patches are automatically grouped together since they are consecutively entered in the sequence of their occurrence in the training image:

location	**..**	**454**	**455**	**456**	**457**	**458**	**..**
patch	..	2 1 1 1 **1** 1 1 1 1	1 1 1 1 **1** 1 2 1 1	2 1 1 2 **1** 1 1 1 1	1 2 1 1 **1** 1 1 1 1	2 1 1 1 **1** 1 2 1 1	..
occurrences	..	28	35	4	1	1	..

For the purpose of better understanding the stages involved in every query of the database are summarized in short:

- Stage 0: Calculate hash index from actual query label patch. The hash value has the property that it is not unique but only very few different patches hash to the same value.
- Stage 1: Choose sub-database according to center pixel of the query label patch.
- Stage 2: From sub-database determine candidate locations in next stage of query for all patches that hash to same hash value.
- Stage 3: Access those locations consecutively and compare query label patch to label patch stored at database location.
- Abort Criterion: If matching label patch found, return corresponding number of occurrences, if no matching patch is found return 0.

8.1.3 Database Generation

The process for the generation of the patch prior database works analogous to the querying of an already computed patch database as presented in the previous chapter.
The patch prior database is generated from all unique patches that are extracted from all training images. This is achieved by first extracting all $h \times w$ patches from each individual training image, where h and w stand for the height and width of the image in pixels respectively. These patches are in the next step reduced to only the unique patches with their corresponding number of occurrences. In the final step the database is updated for each of the unique patches.
Each of the unique patches extracted from the training image is used as input for the query in the database to be built. If the query returns that the query label patch is already present, the corresponding number of its occurrences is increased according to its number of occurrences in the training image. If the query patch can not be found, the patch and its number of occurrences are entered at the end of the sub-database, corresponding to the center pixel of the query patch, and the location information is updated accordingly for the other stages of the query.
The next chapter examines the influence of the patch size on the appearance and complexity of the databases generated in this way.

8.2 Patch Size and Patch Quantity

This chapter discusses the impact of the chosen patch size on the quantity of the unique image patches extracted from the training images, the nature of those patches, and their distribution. It deals with the label configurations of the individual patches, as well as their respective quantities, and how these parameters change when the size of the image patches is varied.
Chapter 8.2.1 deals with the theoretically possible number of patch configurations that could occur. This is followed by an analysis of the circumstances that emerge in practice and how they influence the composition of the databases in Chapter 8.2.2. Finally in Chapters 8.2.3, 8.2.4, and 8.2.5, the actual databases, that were generated for the semantic segmentation process, are examined, regarding the conclusions that can be drawn from the choice of the patch size.

8.2.1 Theoretically Possible Number of Label Configurations

The number of possible label configurations for each patch depends on the patch size and the number of classes in the dataset. The maximum number of possible configurations is calculated with $c^{k \times k}$ where c is the number of semantic classes in the dataset and k is the patch size (assuming the use of quadratic patches). This leads to a very high number of theoretically possible label configurations. For example, assuming the smallest possible patch size of 3×3 and one of the datasets with only 7 semantic categories, the number of possible labelings reaches over 40 millions.
A theoretic example of the 16 possible label configurations ($2^{2 \times 2}$) for a patch of size 2×2 and a theoretical dataset with only 2 different semantic classes is shown in Figure 23.

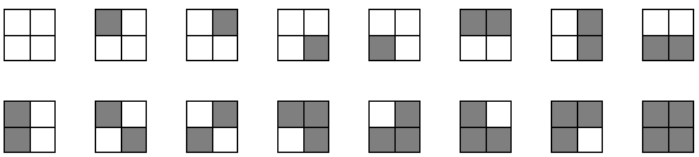

Figure 23: Label configurations for a patch of size 2×2 and 2 different semantic categories.

Note that the label configurations 2 - 5, 6 - 9, 10 - 11, and 12 - 15 in Figure 23 may not be reduced to a single patch respectively, despite their symmetric nature. This is justified by the fact that for example patch 6 in Figure 23 might occur multiple times whereas patch 8 might never occur in the training

images. These patches might correspond to label configurations that denote ground and sky, and in this way correctly mirror the incidence of those label classes in the training images.
This chapter showed the theoretic background concerning the possible number of label configurations. The next chapter discusses the actual distribution of image patches as they occur in the patch prior databases that were generated on the datasets presented in Section 7. This is followed by an examination of those actual databases.

8.2.2 Label Configurations Occurring in Practice

Databases with the very high number of theoretically possible different image patches, as discussed in the previous chapter, will never occur when they are generated from natural images. It is rather unlikely that a patch contains each of the possible semantic categories, in some cases this is even impossible, as in the case of image patches of size 3×3 and a dataset that contains more than nine semantic classes. Most of the training images only contain a small selection of objects of certain classes which limits the number of possible configurations from the outset. The following chapters will show, that only the combination of datasets containing very small images and a large patch size, yields image patches that consist of more than four or five class labels. Another thing that the following chapters will show, especially for the databases built for the smaller patch sizes, is that a rather large percentage of patches is homogenous and only contains a single label. Those patches are usually followed, in terms of quantity, by patches that are extracted at the border between two semantic classes and therefore consist of two different class labels in multiple variations. Thus the actual number of label configurations, that eventually do occur, is strongly reduced in comparison to what would be theoretically possible. This was in fact one of the observations that led to the design of the database in the way that is described in Chapter 8.1, instead of initializing an empty database containing all possible label configurations and incrementing the number of occurrences each time a patch is learned from the training images. This approach would have led to a huge memory overhead since most of the database entries would have remained empty.
The following chapters give a detailed analysis of some aspects of the actual databases generated for the practical part of this master's thesis. For each dataset, the development of the quantity of unique patches for different patch sizes is discussed, and also how these patches are distributed concerning the number of semantic classes they consist of.

8.2.3 Corel-100

The following chapters contain an analysis of the databases generated for each of the datasets presented in Section 7. The patch size is varied between the minimum size of 3×3 and the maximum size of 15×15. The resulting databases are examined concerning the number of unique patches and their total quantitative occurrence.
The Corel-100 dataset (see Chapter 7.1 for details) contains 7 semantic classes, and the training set consists of 50 images. Table 6 shows the numbers of occurrences of individual patches for patch sizes ranging from 3×3 to 15×15. The row 'percentage' holds the fraction of actually occurring patches to the theoretically possible number of patches.

Table 6: Corel-100: Number of unique patches and fraction of actual patches to theoretically possible patches with varying patch size.

patch size	**3 x 3**	**5 x 5**	**7 x 7**	**9 x 9**	**11 x 11**	**13 x 13**	**15 x 15**
quantity	2,044	21,056	73,036	145,746	221,380	292,007	354,890
percentage	0.0051	1.57E-15	2.84E-35	5.14E-62	1.23E-95	4.40E-136	2.53E-183

Table 6 shows that for patch size 3×3 a relatively low quantity of 2,044 individual patches is extracted which corresponds to only $\frac{5}{1{,}000}$ percent of the theoretically possible patches. That number gradually increases up to around 300,000 for the larger patch sizes. While the number of actually occurring unique patches only slowly increases with increasing patch size, the number of theoretically possible label patches increases rapidly. This results in the value for the fraction of actually occurring to theoretically possible patches to rapidly decline. The increase in the number of unique patch occurrences and the parallel decrease in percental occurrences can be seen in Figure 24.

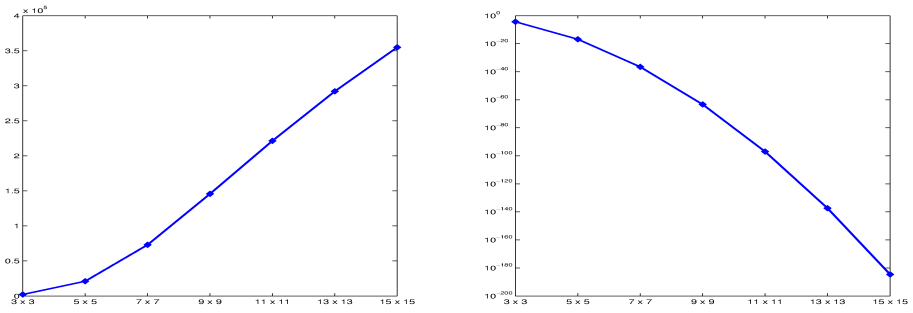

Figure 24: Corel-100: Evolution of unique patch quantity (left) and fraction of actual occurrences to theoretically possible occurrences (right) with varying patch size.

As Figure 24 shows, the unique patch quantity for the Corel-100 dataset starts out low for the minimum patch size before experiencing an almost exponential increase during the following smaller patch sizes. Starting at patch size 7×7 the increase becomes linear and stays that way until the maximum patch size of 15×15 is reached. The plot on the right side shows how the fraction of actually occurring unique patches to theoretically possible patches gradually decreases with growing patch size.
Table 7 shows the distribution of the quantities by breaking the numbers of their occurrences down according to the number of different class labels that the respective label patches consist of. The left column under each column labeled c class(es) contains the number of unique patches and the right column contains the collective number of occurrences of patches with c classes respectively.

Table 7: Corel-100: Distribution of unique patches and their quantity with varying patch size.

	1 class		**2 classes**		**3 classes**		**4 classes**	
3 x 3	7	994,400	1,337	84,614	700	986	0	0
5 x 5	7	916,172	17,211	159,713	3,838	4,115	0	0
7 x 7	7	847,142	63,916	223,594	9,113	9,264	0	0
9 x 9	7	785,754	129,653	278,033	16,086	16,213	0	0
11 x 11	7	730,574	196,535	324,490	24,838	24,936	0	0
13 x 13	7	680,019	256,884	364,797	35,098	35,166	18	18
15 x 15	7	633,211	308,165	400,018	46,614	46,667	104	104

The column marked '1 class' in Table 7 shows that the seven semantic classes in the Corel-100 dataset lead to seven different homogenous label patches that only contain pixels with the same label. The number of their collective occurrences decreases with increasing patch size which means that larger patch sizes lead to less occurrences of homogenous patches. For patches containing two different class labels, the number of individual patches gradually increases from 1337 for patches of size 3×3 to 308,000 for the maximum patch size of 15×15. The values for the number of the total occurrences on the right side of the column labeled '2 classes' are always much higher than the ones on the left which means that many of the individual patches occur multiple times. This is not the case for the columns with three or four different class labels where the values are very close and identical respectively which means that those patches are very rare or even unique. Figure 25 presents the content of Table 7 in the form of a diagram.

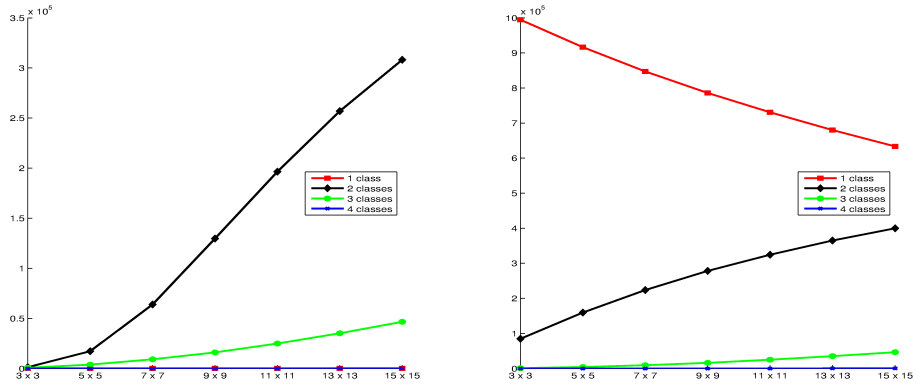

Figure 25: Corel-100: Distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

The left side in Figure 25 shows the development for the individual patches and the right side the development for the total number of those patches in terms of quantity, broken down according to the number of class labels per patch. The plots representing the quantity of patches with one and four different classes stay low for the individual patches while those for two and three different classes increase with different magnitudes. For the total number of patch occurrences on the right side of the diagram, the number of patches with one class decreases in the same manner that the quantity of patches with two different labels increases.
Table 8 and Figure 26 visualize the same information by regarding the same numbers considering their percental distribution.

Table 8: Corel-100: Percental distribution of patches with varying patch size.

	1 class		**2 classes**		**3 classes**		**4 classes**	
3 x 3	0.343	92.07	65.41	7.83	34.25	0.09	0	0
5 x 5	0.033	84.83	81.74	14.79	18.23	0.38	0	0
7 x 7	0.010	78.44	87.51	20.70	12.48	0.86	0	0
9 x 9	0.005	72.76	88.96	25.74	11.04	1.50	0	0
11 x 11	0.003	67.65	88.78	30.05	11.22	2.31	0	0
13 x 13	0.002	62.96	87.97	33.78	12.02	3.26	0.006	0.002
15 x 15	0.002	58.63	86.83	37.04	13.13	4.32	0.029	0.010

Table 8 shows that the percental rate of the unique homogenous patches is negligible while their collective quantity makes up a large portion of all patches, especially for the smaller label patches. The column for two different

classes per label patch shows that those patches make up the better part of the unique patches, and their collective percentage increases at approximately the same rate as that for the homogenous label patches decreases.

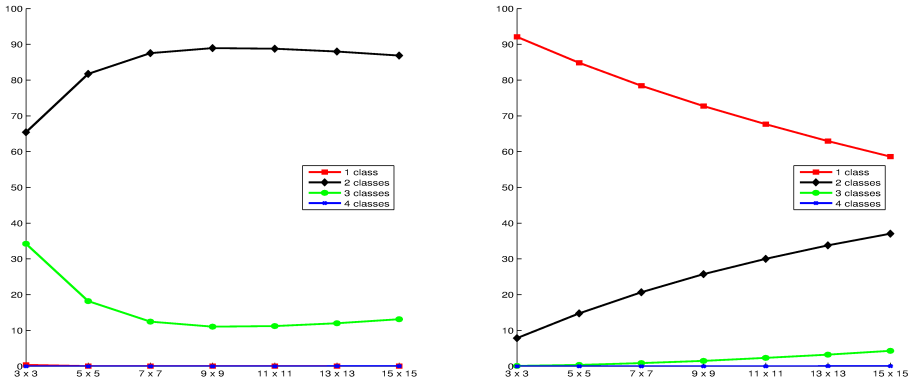

Figure 26: Corel-100: Percental distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

The plots for two and three classes in Figure 26 show a gradual increase and decrease respectively until they reach a value around 90% and 10% respectively. The right hand side shows that the plot for the collective percental distribution is proportional to that of the absolute number of occurrences presented on the right in Figure 25.

8.2.4 MSRC-21

The MSRC-21 dataset contains 21 semantic classes. As was already mentioned in Chapter 7.2, the groundtruth images for this dataset, that are used for the generation of the databases and for the evaluation, are not fully labeled. Therefore the 'void' class is learned as a 22nd semantic category. The databases that are examined in this chapter were built from a training set of 276 images. Table 9 shows the development of the number of individual label patches for the different patch sizes and the percentage of actually occurring patches to theoretically possible patches.

Table 9: MSRC-21: Number of unique patches and fraction of actual patches to theoretically possible patches with varying patch size.

patch size	**3 x 3**	**5 x 5**	**7 x 7**	**9 x 9**	**11 x 11**	**13 x 13**	**15 x 15**
quantity	10,855	95,383	344,317	792,208	1,363,645	1,966,514	2,558,347
percentage	8.99E-7	2.62E-27	5.73E-59	1.45E-101	5.03E-155	2.66E-219	2.31E-294

Resulting from the higher number of semantic classes and training images, the number of different patches for the smallest patch size of 3×3 already totals almost 11,000, and it reaches over 2.5 million different patches for the largest patch size of 15×15. The corresponding fraction of actually occurring unique patches to theoretically possible patches starts resulting from the high number of semantic classes out low with only a fraction of $\frac{9}{10{,}000{,}000}$ percent and rapidly decreases to around 2^{-300} for the maximum patch size of 15×15. Figure 27 shows the increase for the quantities of individual patches and the corresponding decrease for the percental occurrences with increasing patch size.

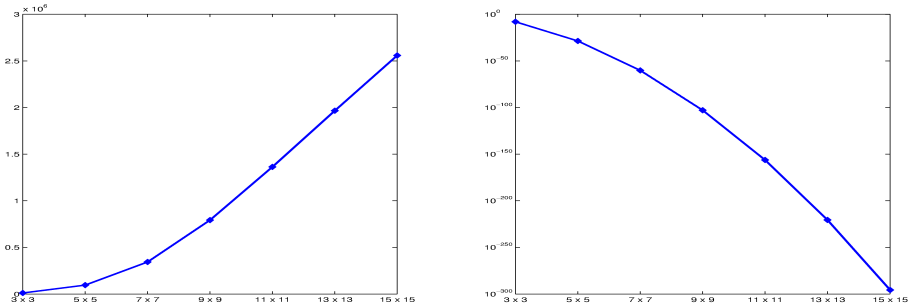

Figure 27: MSRC-21: Evolution of unique patch quantity (left) and fraction of actual occurrences to theoretically possible occurrences (right) with varying patch size.

Again an exponential increase in quantity is visible for the smaller patch sizes of 3×3 through 7×7. After that the increase becomes again linear, as was the case for the Corel-100 dataset. The logarithmic plot on the right side shows the decrease of the fraction of actually occurring to theoretically possible label patches.
Similar to the previous chapter, Table 10 shows the quantity distribution according to the number of semantic class labels per patch.
The 22 different homogenous patches that only consist of a single class label stem from the 21 semantic classes plus the void class. Their total number starts out high at 18.5 million and gradually decreases until it reaches around 15 million patches. For the patches consisting of two different class labels the individual patches start out at around 7,500 for a patch size of 3×3 and reach 2.3 million different individual patches for the maximum patch size of 15×15. The numbers on the right hand side of that column are higher than those on the left hand side, indicating that many of the patches with two different class labels occur multiple times. The number of individual image patches with three different class labels gradually increases while the number

Table 10: MSRC-21: Distribution of unique patches and their quantity with varying patch size.

	1 class		2 classes		3 classes		4 classes		5 classes	
3 x 3	22	18,582,870	7,484	655,273	3,349	4,417	0	0	0	0
5 x 5	22	17,952,274	77,672	1,271,041	17,684	19,240	5	5	0	0
7 x 7	22	17,340,117	302,447	1,858,837	41,774	43,532	74	74	0	0
9 x 9	22	16,748,357	716,072	2,416,078	75,727	77,738	387	387	0	0
11 x 11	22	16,182,277	1,241,602	2,935,707	120,915	123,470	1,106	1,106	0	0
13 x 13	22	15,642,392	1,787,151	3,417,652	176,769	179,944	2,572	2,572	0	0
15 x 15	22	15,124,826	2,311,432	3,867,223	241,807	245,425	5,076	5,076	10	10

of their total occurrences stays similar to that of the individual patches. This means that apart from a few thousand exceptions, most of the patches with three different class labels only occur once. While for the Corel-100 dataset the first patches with four different class labels did not occur before a patch size of 13×13 was reached, for the MSRC-21 dataset the first occurrences of patches with four semantic class labels are already detected at patch size 5×5. At the maximum patch size of 15×15 their number totals around 5000. At that patch size ten patches with even five different class labels are extracted. No patch with four or five different labels occurs more than once throughout the database.

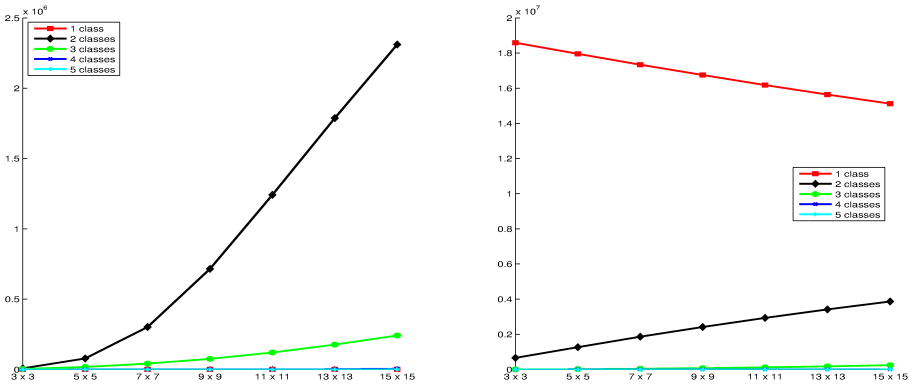

Figure 28: MSRC-21: Distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

The left side of Figure 28 visualizes the progression of the distribution for the individual patches for the varying patch size while the right side illustrates the distribution of the total number of image patches. For the individual patches, the largest increase happens for patches with two different classes,

followed by those consisting of three different labels. For the total patch quantity, the plot corresponding to the homogenous patches decreases in the same manner that the plot denoting patches with two different classes increases. Both diagrams look very similar to the ones presented in the previous chapter in Figure 25 except for the different scale.
Table 11 and Figure 28 give an illustration of the same information by considering the percental distribution.

Table 11: MSRC-21: Percental distribution of patches with varying patch size.

	1 class		**2 classes**		**3 classes**		**4 classes**		**5 classes**	
3 x 3	0.203	96.57	68.95	3.41	30.85	0.02	0	0	0	0
5 x 5	0.023	93.29	81.43	6.61	18.54	0.10	0.005	0.0000	0	0
7 x 7	0.006	90.11	87.84	9.66	12.13	0.23	0.022	0.0004	0	0
9 x 9	0.003	87.04	90.39	12.56	9.56	0.40	0.049	0.0020	0	0
11 x 11	0.002	84.10	91.05	15.26	8.87	0.64	0.081	0.0057	0	0
13 x 13	0.001	81.29	90.88	17.76	8.99	0.94	0.131	0.0134	0	0
15 x 15	0.001	78.60	90.35	20.10	9.45	1.28	0.198	0.0264	0.0004	0.0001

Again the distribution is very similar to the one in Table 8 and Figure 26 for the Corel-100 dataset. Most of the unique patches are accounted for by the patches with two and three labels. Their quantities increase and decrease respectively until they reach values of 90% and 10% respectively.

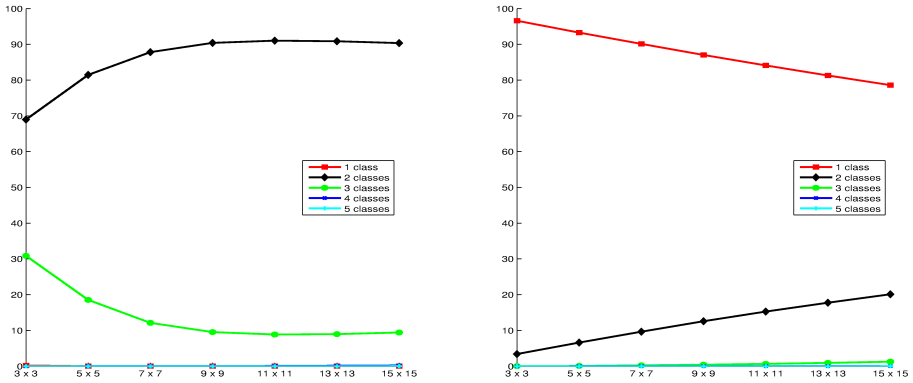

Figure 29: MSRC-21: Percental distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

The plots in Figure 29 visualize the respective increase and decrease for the quantities of label patches with two and three different labels until they reach a value of 90% and 10% respectively. The right side is again proportional to the right side in Figure 28.

8.2.5 Sowerby

The Sowerby dataset (details in Chapter 7.3) contains seven semantic classes, like the Corel-100 dataset, and the training set consists of 52 images. The Sowerby dataset contains the smallest images with a size of only 96×64 pixels. While the databases for the Corel-100 and the MSRC-21 dataset are very similar in terms of evolution of patch quantity and distribution, the databases for the Sowerby dataset are different on those accounts. Table 12 shows the evolution of quantities for the unique patches ranging from size 3×3 through 15×15 and the corresponding percentages.

Table 12: Sowerby: Number of unique patches and fraction of actual patches to theoretically possible patches with varying patch size.

patch size	**3 x 3**	**5 x 5**	**7 x 7**	**9 x 9**	**11 x 11**	**13 x 13**	**15 x 15**
quantity	6,744	39,929	76,995	109,958	137,670	160,531	179,676
percentage	0.0167	2.98E-15	3.00E-35	3.88E-62	7.62E-96	2.42E-136	1.28E-183

The Sowerby dataset contains the same number of semantic classes as the Corel-100 dataset, only 2 training images more, and the images are four times smaller, yet where the Corel-100 database contains only 2,000 different patches for a patch size of 3×3, the corresponding database for the Sowerby dataset already contains 6,744. This corresponds to a percental number of occurrences described by the fraction of actually occurring patches to theoretically possible patches of $\frac{17}{1{,}000}$ percent which is comparatively high. On the other hand the Corel-100 database contains 350,000 different patches of size 15×15, whereas the Sowerby database only holds 180,000 patches for the same patch size. A visualization of the progression of image patch quantities and percental occurrences can be seen in figure 30.

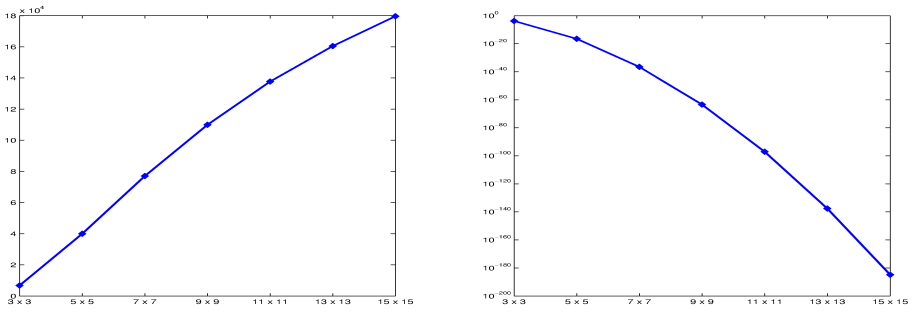

Figure 30: Sowerby: Evolution of unique patch quantity (left) and fraction of actual occurrences to theoretically possible occurrences (right) with varying patch size.

The quantity of the individual patches increases linearly starting at patch size 3×3, until the increase grows less steep at patch size 9×9. The fraction of actually occurring label patches to theoretically possible number of patches for the corresponding patch size shows a very similar decrease to that shown by the other two datasets.
Table 13 once again shows the quantity distribution, broken down by the number of different semantic categories per patch. Where even the MSRC-21 dataset with its 21 plus 1 semantic classes only led to patches with at most five different labels, the Sowerby dataset with its smaller images yields patches that consist of all seven possible class labels.

Table 13: Sowerby: Distribution of unique patches and their quantity with varying patch size.

	1 class		**2 classes**		**3 classes**		**4 classes**	
3 x 3	7	263,336	2,352	49,753	3,761	5,756	589	608
5 x 5	7	226,197	21,213	73,222	14,775	16,117	3,376	3,394
7 x 7	6	197,135	41,614	86,258	25,371	26,079	7,924	7,936
9 x 9	6	173,218	58,014	93,914	33,530	33,941	13,417	13,424
11 x 11	6	153,127	69,567	97,949	39,840	40,149	19,050	19,056
13 x 13	6	135,969	77,118	99,852	44,411	44,665	24,432	24,438
15 x 15	6	121,019	81,421	99,969	48,095	48,342	29,440	29,444
	5 classes		**6 classes**		**7 classes**			
3 x 3	33	33	2	2	0	0		
5 x 5	500	500	58	58	0	0		
7 x 7	1,817	1,817	257	257	6	6		
9 x 9	4,250	4,250	712	712	29	29		
11 x 11	7,512	7,512	1,592	1,592	103	103		
13 x 13	11,225	11,225	3,054	3,054	285	285		
15 x 15	15,111	15,111	4,970	4,970	633	633		

The first thing that is observable in Table 13 is that already for patches of size 7×7 six patches do occur that contain each of the possible seven semantic classes. At the same patch size it is no longer possible to extract all seven homogenous patches. This fact is originated by the semantic class 'road marking' that composes only tiny structures in the small images. Like for the Corel-100 and MSRC-21 datasets, the absolute number of homogenous patches gradually decreases, although a little faster than is the case for the other two datasets. Another thing that is similar to the other two datasets is

that the highest quantity of individual image patches is extracted for patches with two different class labels. Their total number of occurrences starts out at around 50,000 patches for patch size 3×3, reaches 73,000 at patch size 5×5, and levels between 90,000 and 100,000 patches for the patch sizes of 7×7 through 15×15. The comparison between left and right hand side of the column marked '2 classes' indicates that for the smaller patch sizes many duplicates do occur, while for the larger patch sizes the variety decreases. The patches with three, four, five, and even six different classes show a relatively strong increase in terms of quantity as the patch size is increased. In those cases most of the patches extracted only occur a single time in the database.

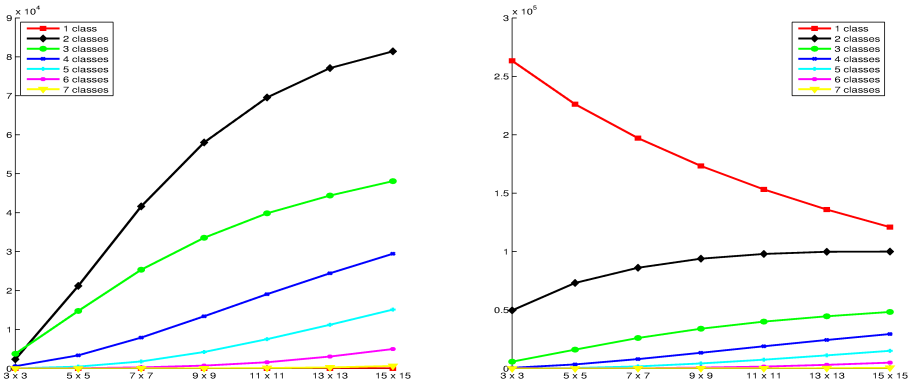

Figure 31: Sowerby: Distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

Figure 31 shows a visualization of Table 13. The highest increase in quantity is observed for individual patches with two different classes, but as noted in the discussion of Table 13, the individual patches for three through six different labels also show a relatively high increase when compared to similar patches for the two other datasets. The diagram on the right denoting the total number of patch occurrences shows a rapid decline for the homogenous patches and a slow gradual increase for patches with more labels.
Table 14 and Figure 32 show the same distribution data for the different label patches from a percental point of view. Again the homogenous label patches make up a negligible portion of the unique patches and the better part of the unique patches is made up by the unique label patches consisting of two and three classes.
The right part of Figure 32 is again proportional to the right part of Figure 31 as was also the case for the other datasets. One thing that can be observed when looking at the left part of the figure, denoting the percent-wise quantity of the individual patches, is that the patches with three different class

Table 14: Sowerby: Percental distribution of patches with varying patch size.

	1 class		**2 classes**		**3 classes**		**4 classes**	
3 x 3	0.104	82.42	34.88	15.57	55.77	1.80	8.73	0.19
5 x 5	0.018	70.80	53.13	22.92	37.00	5.04	8.46	1.06
7 x 7	0.008	61.70	54.05	27.00	32.95	8.16	10.29	2.48
9 x 9	0.006	54.22	52.76	29.40	30.49	10.62	12.20	4.20
11 x 11	0.004	47.93	50.53	30.66	28.94	12.57	13.84	5.96
13 x 13	0.004	42.56	48.04	31.25	27.67	13.98	15.22	7.65
15 x 15	0.003	37.88	45.32	31.29	26.77	15.13	16.39	9.22
	5 classes		**6 classes**		**7 classes**			
3 x 3	0.49	0.01	0.03	0.00	0	0		
5 x 5	1.25	0.16	0.15	0.02	0	0		
7 x 7	2.36	0.57	0.33	0.08	0.01	0.002		
9 x 9	3.87	1.33	0.65	0.22	0.03	0.009		
11 x 11	5.46	2.35	1.16	0.50	0.07	0.032		
13 x 13	6.99	3.51	1.90	0.96	0.18	0.089		
15 x 15	8.41	4.73	2.77	1.56	0.35	0.198		

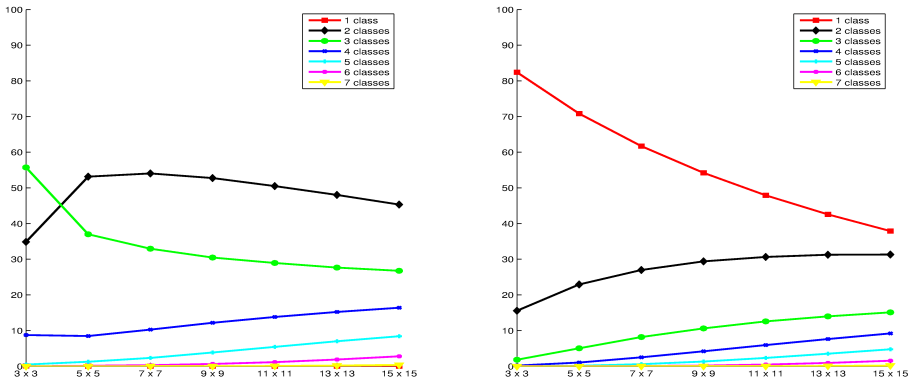

Figure 32: Sowerby: Percental distribution of patches with varying patch size, left: unique patches, right: quantity of patches.

labels have a higher percentage than those with only two labels for patch size 3×3 before declining and staying roughly at 30% while the patches with two different classes increase to roughly 50% and stay there for patches of size 5×5 and larger.

This section presented the design details of the database that was implemented to learn the patch prior knowledge. It also described how the database

is queried and how it is generated. After a discussion of the theoretically possible number of unique label patches and arguments why this high number will never be reached with natural images, the actual databases that were generated for the experiments were examined in detail. For the three datasets introduced in Section 7 patch prior databases for label patches of size 3×3 to 15×15 were built and the resulting databases were analyzed in detail. The influence of the patch size on the number of patches with different quantity of classes was examined in terms of absolute numbers as well as for the percental distribution.
The next section contains the actual experiments that were conducted on the three datasets by utilizing the databases generated for the different patch sizes.

9 Experiments

This section presents the results of the experiments that were conducted on the image datasets presented in Section 7. It shows the influence of the choice of parameters like patch size, unary potentials or the weights determining the relative importance of the potentials on the segmentation results. Chapter 9.1 describes the means of the evaluation with the evaluation parameters, the form the results are presented in, and the evaluation framework itself. The following Chapters 9.2, 9.3, and 9.4 deal with the outcomes of the experiments that were conducted on the individual datasets and presents the results in the form of tables and diagrams along with corresponding interpretations.

9.1 Evaluation

This chapter presents the means of evaluation that were taken to quantify the segmentation results that could be achieved with the semantic segmentation framework presented in Section 6. The first chapter presents the parameters that are used to measure the quality of the segmentation results. Then the evaluation framework, that was used to automatically evaluate the segmentation results, is described. Finally, the different methods of comparison that were used to rate the achieved results and compare them to the state-of-the-art methods are discussed.

9.1.1 Evaluation Parameters

This chapter presents the parameters that are used in the process of evaluating the system developed in the course of this master's thesis.
Different literature uses different parameters to depict the quality of the semantic segmentation results. The most commonly used measures are the recall and the intersection vs. union measure:
Recall (also termed labeling accuracy, average pixel-wise accuracy, or average pixel-wise recognition rate): This measure denotes the percentage of correctly labeled pixels:

$$\frac{TP}{TP + FN} \tag{8}$$

Intersection vs. Union: This measure also takes pixels into account that were classified incorrectly:

$$\frac{TP}{TP + FN + FP} \tag{9}$$

The variables TP, FP, and FN denote pixels classified as true positive, false positive, and false negative respectively.

The segmentation accuracy is divided into two categories that are termed with different names in the literature. The global level captures the percentage of image pixels that were classified correctly, and is called pixel accuracy or global accuracy. The average level captures the average classification accuracy per semantic category and is commonly referred to as class accuracy.

9.1.2 Evaluation Framework

The evaluation framework used in this thesis compares the output of the semantic segmentation framework to the desired segmentation results represented by the annotated groundtruth. The scores that rate the quality of the segmentation results are calculated according to the intersection vs. union measure from Equation 9 for each of the segmented images. The output is presented in the form of the average segmentation accuracy for each of the semantic classes contained in the respective dataset, as well as the pixel or global accuracy and the class or average accuracy.

9.1.3 Segmentation Accuracy and Patch Size

The main focus of the analysis of the results lies on the influence of the patch size on the segmentation accuracy. The evaluation framework described in the previous chapter rates the segmentation results that were calculated by the semantic segmentation framework described in Section 6. The evaluation process is repeated for various patch sizes and different weight parameters, and each of the following chapters presents the results in the form of tables and diagrams that display the segmentation accuracy versus the patch size. The patch sizes range from the smallest possible size of 3×3 to a maximum size of 15×15 for all datasets. The tables and diagrams present the evaluation results for two different sets of weight parameters that discern the respective importance of the unary and pairwise potentials, as discussed in Chapter 3.3. The weight for the unary potentials is kept constant at a relatively high value of $\alpha = 500$ for all the experiments to emphasize the importance of the unary potentials that are manipulated to achieve the refined semantic segmentation results. The influence of the parameter for the pairwise potentials is examined by varying it between a very low parameter of $\beta = 10$ and a slightly higher value of $\beta = 50$. If the results achieved with different weight parameters are very similar, only the superior results are presented and the rest is omitted. The beginning of each chapter dealing with one of the datasets presented in Section 7 includes for comparison purposes a table holding the segmentation accuracies of the maximum likelihood estimation (MLE) on the unary potentials and of a single graph cuts

regularization that takes the unaltered unary potentials as input. All tables and diagrams present the segmentation accuracies in percent.

9.1.4 Confusion Matrix

The segmentation results are also presented in the form of a confusion matrix of object classes vs. object classes. This type of matrix gives an indication of which semantic classes are classified correctly most frequently and which classes can easily be confused with other classes. The presented confusion matrixes capture the segmentation accuracies achieved with the parameter set that leads to the highest rating in the evaluation process. The diagonal elements of the square matrix contain the percentage of the correctly labeled classes. The values are identical with the corresponding column in the corresponding table. The non-diagonal elements of each row capture the amount of confusion with the respective class denoted by the column. Table 15 shows an example of a fictitious confusion matrix with three different classes.

Table 15: Example confusion matrix.

	Class A	**Class B**	**Class C**
Class A	**80**	20	
Class B	5	**70**	25
Class C			**100**

In this example, class A is correctly classified in 80% of all cases, and every fifth time it is incorrectly classified as class B. Class B is correctly classified in only 70% of all cases. 5% of all instances of class B are confused as being of class A, and every fourth time class B is confused as being of class C. Class C is always classified correctly in this example.
All tables representing the confusion matrixes present the amount of correctly classified or confused pixels in percent.

9.1.5 Example Segmentation Results

Each of the chapters dealing with the evaluation results of the respective datasets is concluded by a number of exemplary segmentation results illustrating the strengths and weaknesses of the developed semantic segmentation framework.

9.2 Corel-100

This chapter presents the semantic segmentation results that can be achieved with the implemented semantic segmentation framework on the Corel-100 dataset introduced in Chapter 7.1. The outcomes are presented in the form of tables and diagrams along with a discussion of the results.

9.2.1 Segmentation Accuracy and Patch Size

The first set of results that this chapter presents, consists of some baseline segmentation accuracies. As mentioned in Chapter 6.2 the MLE is the basis for the manipulation of the unary potentials in the two-step iterative process, so the corresponding segmentation accuracy establishes a baseline that should be exceeded by the developed method. Another goal is to improve the segmentation results by iteratively applying the graph cuts approach to the modified potentials. For that reason Table 16 holds the segmentation accuracies for the MLE on the unary potentials and for a single graph cuts regularization step on the potentials with the weight parameters that are used in the further course of this chapter. The potentials for the Corel-100 dataset were calculated with the Automatic Labelling Environment.

Table 16: Corel-100: Segmentation accuracies for MLE and single graph cuts regularizations on unary potentials, potentials calculated with ALE.

	MLE	**Graph Cuts** $\alpha = 500, \beta = 10$	**Graph Cuts** $\alpha = 500, \beta = 50$
rhino/hippo	78.07	75.89	41.62
polar bear	72.42	49.38	41.10
water	89.34	96.12	93.67
snow	78.20	79.27	77.85
vegetation	65.84	71.85	60.52
ground	56.72	48.90	46.73
sky	28.87	26.50	25.31
class accuracy	67.07	63.99	55.26
pixel accuracy	71.79	71.26	62.93

Table 16 shows the limitations of the potentials calculated with the ALE. Where the results of a single graph cuts regularization step should lead to an improvement for the segmentation accuracy compared to a simple MLE on the potentials, in the case of the potentials calculated with the ALE, applying graph cuts always leads to a decrease in segmentation accuracy. This decrease

is rather small for the pixel accuracy and a low value for parameter β that specifies the weight of the pairwise potentials, but already reaches 3% for the class accuracy. For a higher value of β the accuracy decline is even worse. Keeping these accuracy values in mind as baselines, Table 17 presents the segmentation accuracies for a varying patch size while keeping the weight values for the potentials constant.

Table 17: Corel-100: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 10$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
rhino/hippo	79.66	79.68	79.71	79.55	79.49	78.89	76.95
polar bear	73.68	75.29	76.18	76.93	77.76	76.45	77.63
water	91.62	92.27	93.20	93.68	93.89	94.79	92.03
snow	79.26	79.56	80.18	81.56	81.28	81.29	81.40
vegetation	67.86	68.30	69.02	69.78	70.78	70.91	70.86
ground	56.83	56.52	56.59	56.08	55.88	55.78	55.45
sky	28.64	28.92	29.82	29.43	29.09	29.02	26.99
class accuracy	68.22	68.65	69.24	69.57	**69.74**	69.59	68.76
pixel accuracy	73.15	73.48	74.03	74.46	**74.66**	74.68	73.90

Table 17 shows a gradual increase of the segmentation accuracy for most of the seven classes with increasing patch size. The accuracies gradually improve until a patch size of 11×11 is reached, after which the values start to decline again. Since this increase and the following decrease is almost identical for the respective label classes, the resulting average class and pixel accuracies show an identical ascent and descent for patch sizes 3×3 through 11×11. In both cases the accuracy gain is about 1.5%. While the pixel accuracy stays almost constant for patch size 13×13, the class accuracy already gets worse again for this patch size. For image patches of size 15×15 the values of both average pixel and class accuracy decrease to similar values as achieved with patch sizes 5×5 and 7×7. The highest segmentation score in terms of average class accuracy and average pixel accuracy is achieved with patch size 11×11 for the parameter set of $\alpha = 500$ and $\beta = 10$.
Figure 33 illustrates the evolution of the average class accuracy (blue) and the average pixel accuracy (red) in the form of a diagram. The two plots proceed in almost parallel over the whole course of the diagram. The plot shows the gradual parallel accuracy gain until a patch size of 11×11 is reached. For patch size 13×13 the average class accuracy then starts to decline while the

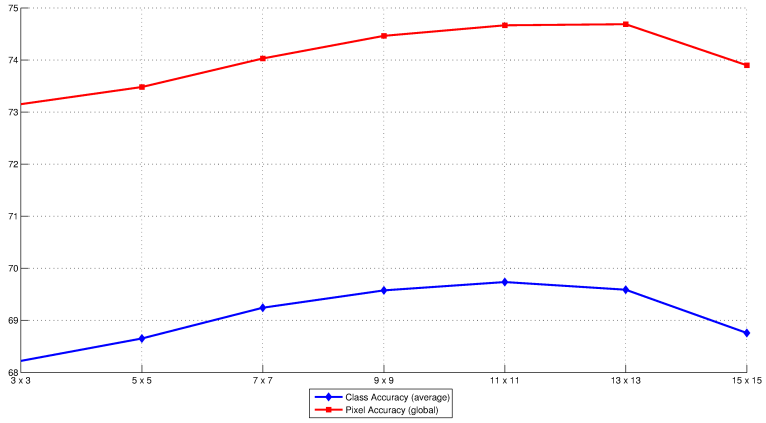

Figure 33: Corel-100: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 10$.

average pixel accuracy stays constant for that patch size. Finally, the values for both class and pixel accuracy distinctly fall for the maximum patch size of 15×15.
Table 18 shows the segmentation accuracies for the various classes and different patch sizes similar to Table 17 only with a slightly higher weight of $\beta = 50$ for the pairwise potentials.

Table 18: Corel-100: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
rhino/hippo	81.81	79.79	80.41	80.65	79.13	78.70	77.49
polar bear	79.23	79.22	79.51	77.62	77.58	77.66	72.95
water	94.74	94.80	95.18	92.19	92.28	92.33	90.00
snow	76.57	77.15	78.26	78.62	78.35	78.48	78.88
vegetation	69.63	70.33	70.98	70.97	71.92	70.49	71.10
ground	55.90	55.58	55.39	55.80	56.09	55.28	54.44
sky	27.07	28.12	28.44	28.56	27.88	27.58	24.78
class accuracy	69.28	69.28	**69.74**	69.20	69.03	68.65	67.09
pixel accuracy	74.01	74.00	**74.49**	73.98	74.00	73.52	72.60

With the altered parameter set the different classes show a different evolution of their respective segmentation accuracies. Table 18 shows a gradual increase in segmentation accuracy for class 'snow' while for example the class

'rhino/hippo' starts out with a high accuracy of 81.81% for patch size 3×3 before dropping to 79.79% for patch size 5×5 and then again improving to 80.41% for patch size 7×7. This behavior of the different classes is mirrored by a resulting average class and pixel accuracy that stay practically constant over the course of the patch sizes 3×3 through 11×11 with peaking accuracies for image patches of size 7×7. The larger patch sizes of 13×13 and 15×15 again lead to a strong decline in both pixel and class accuracy.

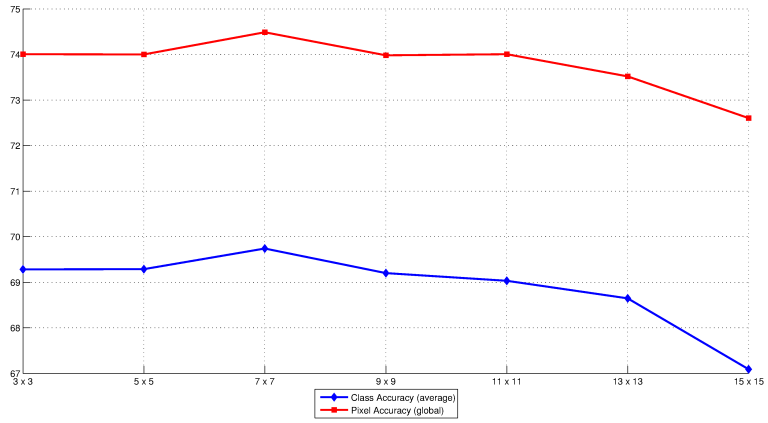

Figure 34: Corel-100: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

Figure 34 shows the evolution of average class accuracy (blue) and average pixel accuracy (red) that is represented by the values in Table 18. The plot illustrates the almost unvarying accuracies for the smaller patch sizes with a peak for size 7×7, followed by a strong decline for the larger patches.

9.2.2 Confusion Matrix

This chapter presents the outcome of the experiments achieved with the parameter set leading to the highest segmentation score. The parameters are the usual $\alpha = 500$ for the weight of the unary potentials and the low value of $\beta = 10$ for the pairwise potentials. The best results can be achieved for patch size 11×11 for the Corel-100 dataset.
The diagonal of the confusion matrix holds the percentage values of the correctly labeled pixels. The values are identical to the corresponding column in Table 17. The matrix shows that the class with the highest segmentation accuracy is 'water' with a value of almost 94%. This class is most frequently confused with 'ground' (3.76%) and sometimes with 'rhino/hippo', 'vegetation', and 'snow'. The class 'rhino/hippo' that can achieve an accuracy of about 80% has the highest confusion value of 13.57% with 'water' followed

Table 19: Corel-100: Confusion matrix for patch size 11×11, $\alpha = 500$, $\beta = 10$.

	rhino/hippo	**polar bear**	**water**	**snow**	**vegetation**	**ground**	**sky**
rhino/hippo	**79.49**		13.57		4.54	2.41	
polar bear	4.52	**77.76**		16.32	0.27	1.13	
water	0.90		**93.89**	0.47	0.97	3.76	
snow	0.35	8.91	0.98	**81.28**	1.33	7.14	
vegetation	2.84	0.85	0.47	8.76	**70.78**	14.81	1.50
ground	10.31	0.88	4.67	9.34	18.93	**55.88**	
sky	0.59			16.50	11.57	42.26	**29.09**

by 4.54% for 'vegetation' and 2.41% for 'ground'. This makes sense, since for the Corel-100 dataset in most images of hippos the hippo is surrounded by water, and the rhino images contain rhinos in front of vegetation standing on the ground. For the same reason the class 'polar bear' with an average class accuracy of 77.76% gets most frequently confused with 'snow' (16.32%). The class 'sky' achieves with only 29.09% the lowest overall segmentation results which is even lower than its confusion index with the class 'ground' which has a value of 42.26%. This poor performance is based on the fact that the dataset only contains very few images with pixels of class 'sky', and in the few cases that do occur, the 'sky' areas are mostly very small. In most cases the areas consisting of 'sky' pixels make up only a small part at the top of the image and frequently get eliminated by the graph cuts regularization. The next chapter presents some example segmentation results.

9.2.3 Example Segmentation Results

This final chapter for the evaluation of the Corel-100 dataset presents some exemplary semantic segmentation results that could be achieved with the presented method. Figure 35 shows eight examples with the original image at the top row, the desired result consisting of the labeled groundtruth in the middle row, and the segmentation results that were computed with the implemented method for the parameter set of $\alpha = 500$, $\beta = 10$, and a patch size of 11×11 at the bottom row.

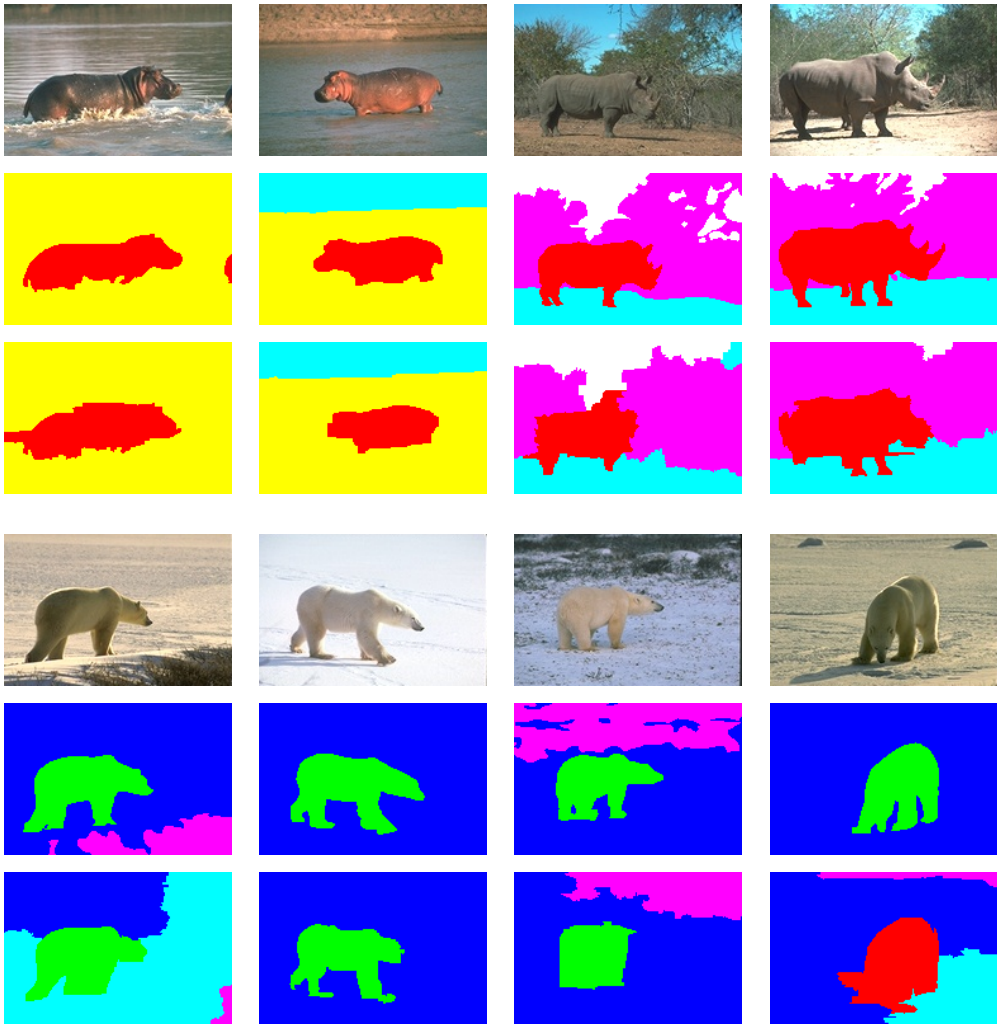

Figure 35: Corel-100: Example images, corresponding groundtruth, and achieved segmentation results (top to bottom, respectively) for patch size 11×11, $\alpha = 500$, and $\beta = 10$.

The examples presented in Figure 35 show images of hippos (top left two) and rhinos (top right two) and polar bears (bottom). The results show a high level of similarity with the groundtruth in most of the cases. For the polar bear at the bottom left the background is not always classified correctly and the polar bear at the bottom right is incorrectly classified as being of class 'rhino/hippo'.

9.2.4 Summary

The results presented in this chapter show that the implemented method can yield good segmentation results for the Corel-100 dataset depending on the choice of parameters. Table 20 shows a comparison of the accuracies reported by the state-of-the-art methods with the best results achieved with the presented method.

Table 20: Corel-100: Comparison of achieved results with state-of-the-art.

Method	**accuracy**	
	class	**pixel**
Shotton et al. [42]	-	74.6%
He et al. [19]	-	80.0%
Toyoda & Hasegawa [46]	-	83.0%
Method	69.74%	74.66%

Even though the presented method can with an average class accuracy of 69.74% and an average pixel accuracy of 74.66% not exceed the results reported by the recent state-of-the-art methods of accuracies ranging from 74.6% up to 83%, the achieved segmentation accuracy exceeds the baseline accuracies presented in Table 16 for all tested patch sizes. The results also show that an increase in segmentation accuracy can be reached for increasing patch sizes up to a certain point where the accuracy starts declining again. Depending on the chosen set of weights for the unary and pairwise potentials, using patches beyond a certain size becomes counterproductive. The example segmentation results presented in Figure 35 show the high level of similarity with the desired results represented by the manually labeled groundtruth that the implemented method can achieve.

9.3 MSRC-21

This chapter presents the semantic segmentation results that can be achieved with the presented semantic segmentation framework on the MSRC-21 dataset discussed in Chapter 7.2. The outcomes of the experiments are presented in the form of tables and diagrams along with a discussion of the results.
The MSRC-21 dataset is the only dataset used in this master's thesis for which two sets of unary potentials are available. The first one consists of the potentials calculated with the Automatic Labelling Environment that is also used for each of the other datasets. The other potentials were calculated for the use in [25] and are also used in parts of this evaluation to attain a valid comparison to this state-of-the-art method.
The following chapter presents the results of the evaluation with a setup that is identical to the one reported in [25]. Then Chapter 9.3.2 examines the influence of the different unary potentials on the segmentation results. This is followed, similar to the composition of the other chapters dealing with the experimental results, by the discussion of the confusion matrix generated with the best set of parameters and some exemplary segmentation results.

9.3.1 Segmentation Accuracy and Patch Size

Table 21 again establishes baseline segmentation accuracies for the evaluations following here. The experiments presented in this chapter were conducted on the same split into training and test data as used in [25]. Additionally to the two sets of unary potentials, there are also two different sets of groundtruth data available for the MSRC-21 dataset as described in Chapter 7.2. As already discussed there, the groundtruth used in [25] presents the problem that a lot of pixels are labeled as 'void', especially in the areas of transitions between two different label classes. Since a lot of the improvement achieved with the implemented method happens in these areas, these improvements are often not considered in the evaluation on this groundtruth. For that reason the evaluation of the segmentation results is conducted one time on the groundtruth with the many 'void' pixels to reproduce the setup that was used in [25], and a second time with the relabeled groundtruth to notice a potential improvement in the otherwise unlabeled areas. Since the groundtruth with the unlabeled areas is used in the state-of-the-art paper of Krähenbühl and Koltun [25], this groundtruth will in the following be referred to as state-of-the-art groundtruth.
The parameter sets for the weights specifying the relative importance of the unary and pairwise potentials were again chosen with $\alpha = 500$ and $\beta = 10$ and $\beta = 50$ respectively.

Table 21: MSRC-21: Evaluation results for MLE and single graph cuts regularizations on unary potentials, potentials as used in [25], two different sets of groundtruth.

	groundtruth as used in [25]			relabeled groundtruth		
	MLE	**GC:** $\alpha = 500$, $\beta = 10$	**GC:** $\alpha = 500$, $\beta = 50$	**MLE**	**GC:** $\alpha = 500$, $\beta = 10$	**GC:** $\alpha = 500$, $\beta = 50$
building	71.97	72.21	72.53	71.03	72.71	73.51
grass	98.18	98.47	98.62	88.50	89.15	90.02
tree	89.72	88.28	89.01	83.88	83.61	84.85
cow	84.33	82.65	81.90	88.10	88.71	89.36
sheep	80.57	81.38	81.80	82.97	84.86	86.04
sky	93.36	95.79	96.09	83.15	89.63	90.63
aeroplane	82.47	81.27	80.41	85.16	85.99	85.50
water	67.53	69.83	69.97	62.83	65.82	65.51
face	88.16	88.30	89.09	86.30	87.49	88.65
car	84.23	85.25	85.14	85.93	87.61	87.38
bicycle	91.15	90.24	89.55	91.45	91.34	91.01
flower	90.75	91.81	93.08	90.62	91.87	92.92
sign	70.00	76.08	77.89	71.17	78.94	80.99
bird	47.64	46.73	45.38	47.94	48.87	47.77
book	94.10	94.78	95.08	93.97	94.57	94.89
chair	59.33	60.04	59.82	61.07	62.11	62.37
road	88.80	89.30	89.11	82.43	83.83	83.92
cat	75.75	79.19	79.68	75.99	79.61	80.09
dog	46.05	44.19	46.21	46.32	44.53	46.58
body	79.94	81.84	80.29	79.41	82.53	81.09
boat	25.16	25.64	22.94	26.42	27.36	24.57
class accuracy	76.63	77.30	77.31	75.46	77.20	77.51
pixel accuracy	84.05	84.75	84.95	80.57	82.39	82.98

Table 21 shows the influence that the different sets of groundtruth images have on the outcome of the evaluation. The two evaluations were conducted on the completely identical unary potentials and the completely identical output results from the graph cuts framework respectively. For the MLE on the unary potentials the average class accuracy is 1% higher and the average pixel accuracy is even 3.5% higher for the state-of-the-art groundtruth com-

pared to the relabeled groundtruth. Table 21 also shows the strength of the potentials used in [25] which were calculated with the TextonBoost framework [42] in comparison to the potentials calculated with the ALE. Where the application of a single graph cuts regularization step on the unmodified ALE potentials leads to a decline in segmentation accuracy, the better potentials used in [25] lead to an accuracy increase of two and more percent compared to the MLE on the unary potentials.
Table 22 presents the results of the semantic segmentation with the same split into training and test data as used in [25], evaluated on the groundtruth also used there.

Table 22: MSRC-21: Evaluation results for varying patch size, potentials and groundtruth as used in [25], $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
building	71.53	71.68	71.79	71.93	71.97	72.15	72.19
grass	98.62	98.62	98.68	98.80	98.80	98.85	99.00
tree	86.76	86.64	86.13	85.76	85.80	85.77	85.38
cow	80.72	80.93	80.95	80.74	80.34	79.74	79.43
sheep	79.53	79.31	79.10	78.70	79.12	78.78	79.12
sky	95.97	95.96	95.96	96.10	96.10	96.07	96.24
aeroplane	80.08	79.76	79.66	79.05	78.79	78.05	77.15
water	70.20	70.41	70.62	70.96	70.82	70.91	71.01
face	88.61	88.47	88.22	87.67	87.73	87.22	85.94
car	85.41	85.35	85.03	84.96	84.14	84.46	83.74
bicycle	89.06	89.33	89.28	88.76	88.02	87.30	86.34
flower	92.66	92.93	93.15	93.16	92.97	93.44	93.27
sign	76.42	76.12	76.92	78.86	79.07	78.76	78.62
bird	46.92	47.26	46.02	45.26	44.70	44.66	44.62
book	94.99	94.99	95.18	95.27	95.40	95.45	95.45
chair	59.62	59.33	59.20	59.25	58.77	57.44	58.16
road	89.78	89.85	89.98	89.96	90.05	89.88	89.93
cat	78.10	78.47	79.29	79.33	78.02	78.02	78.64
dog	48.91	49.77	50.60	51.20	49.39	48.15	47.75
body	77.14	79.74	79.45	77.78	77.63	77.56	77.19
boat	22.28	21.06	20.55	18.19	17.59	16.94	15.37
class accuracy	76.82	76.95	**76.94**	76.75	76.44	76.17	75.93
pixel accuracy	84.54	84.62	**84.65**	84.66	84.54	84.45	84.38

The 21 different classes in the MSRC-21 dataset show different developments over the course of the increasing patch sizes. While classes like 'building', 'grass', or 'water' experience a gain in accuracy with growing patch size, other classes like 'tree', 'car', or 'boat' at the same time show an accuracy decline. This results in an average pixel accuracy that stagnates at around 84.5% and an average class accuracy that slowly degenerates from 77% down to 76% for the evolution from small to large patch sizes.
The average class and pixel accuracies are also illustrated in the plots of Figure 36.

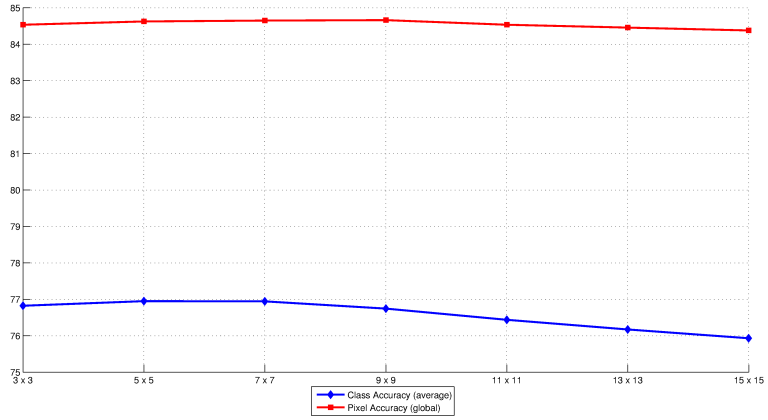

Figure 36: MSRC-21: Evaluation results for varying patch size, potentials and groundtruth as used in [25], $\alpha = 500$, $\beta = 50$.

Figure 36 shows the average pixel accuracy (red) stagnating at around 84.5% and the average class accuracy (blue) with a slight accuracy gain for patches of sizes 5×5 and 7×7 followed by a decline of 1% until the maximum patch size of 15×15 is reached.
Since other configurations of weight parameters lead to very similar segmentation results, these results are omitted.
As Table 22 and Figure 36 presenting the outcome of the evaluation on the results achieved with weight parameters $\alpha = 500$ and $\beta = 50$ presented, the evaluation with the state-of-the-art groundtruth showed no improvement of the segmentation accuracy, neither for the average class accuracy nor for the average pixel accuracy.
Table 23 shows the outcome of the evaluation on the exactly identical results as those used for the generation of Table 22 but this time the relabeled groundtruth is used.
Classes like 'building', 'grass', or 'water', that gained accuracy with growing patch size in Table 22 also show this behavior in Table 23, while classes like

Table 23: MSRC-21: Evaluation results for varying patch size, potentials as used in [25], relabeled groundtruth, $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
building	72.10	72.26	72.45	72.84	72.93	73.24	73.31
grass	90.13	90.20	90.45	90.81	91.02	91.20	91.31
tree	82.56	82.50	82.15	81.85	81.90	81.88	81.34
cow	88.04	88.35	88.60	88.54	88.28	87.80	87.75
sheep	83.68	83.47	83.42	83.26	83.94	83.72	84.07
sky	90.56	90.60	90.70	91.01	91.22	91.23	91.41
aeroplane	85.08	84.98	84.94	84.26	83.89	83.48	82.60
water	65.70	66.00	66.27	66.81	66.80	66.95	67.15
face	88.05	87.98	87.67	87.25	87.31	86.63	85.32
car	87.53	87.50	87.25	87.24	86.42	86.76	85.99
bicycle	90.48	90.68	90.79	90.40	89.90	89.39	88.48
flower	92.65	92.94	93.18	93.22	93.04	93.54	93.45
sign	79.39	79.03	79.97	82.07	82.28	82.19	82.11
bird	48.67	49.03	47.75	47.00	46.48	46.53	46.51
book	94.76	94.78	94.99	95.10	95.27	95.32	95.32
chair	61.89	61.73	61.82	62.11	61.95	60.98	61.83
road	84.87	84.98	85.19	85.34	85.55	85.51	85.59
cat	78.50	78.85	79.73	79.74	78.47	78.59	79.17
dog	49.31	50.18	51.01	51.62	49.82	48.57	48.17
body	77.96	80.64	80.39	78.80	78.74	78.71	78.34
boat	23.78	22.47	21.96	19.46	18.90	18.18	16.50
class accuracy	76.94	77.10	77.18	**77.08**	76.86	76.68	76.46
pixel accuracy	82.60	82.73	82.87	**83.05**	83.07	83.08	83.03

'tree', 'car', or 'boat' again show a decline in class accuracy. This results in a slightly higher average class accuracy for the best parameter configuration compared to the other evaluation. The pixel accuracy that stagnated in the case of the evaluation with the state-of-the-art groundtruth, now shows the desired accuracy gain with increasing patch size, at least for patch sizes 3×3 through 9×9 after which the average pixel accuracy stays at around 83%. Where in the case of the evaluation on the other groundtruth neither the average pixel accuracy nor the average class accuracy could exceed the baselines established in Table 21 the evaluation with the relabeled groundtruth leads for both accuracies to higher values than the baseline values.

Figure 37 visualizes the evolution of the average class accuracy and the average pixel accuracy over the course of the increasing patch size.

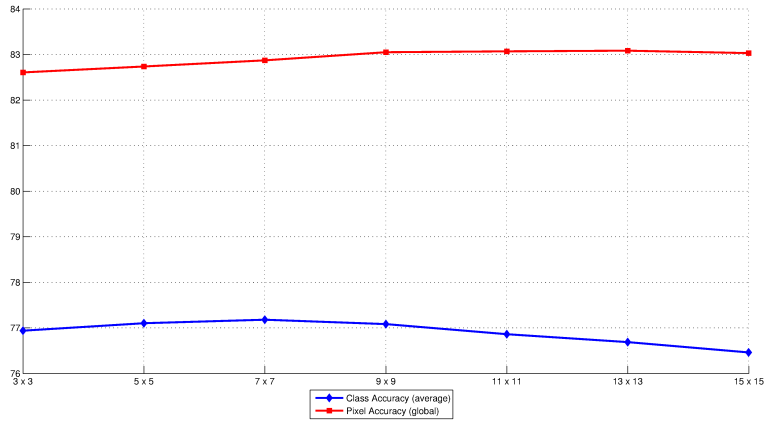

Figure 37: MSRC-21: Evaluation results for varying patch size, potentials as used in [25], relabeled groundtruth, $\alpha = 500$, $\beta = 50$.

The plots in Figure 37 show that the average class accuracy (blue) is very similar in magnitude and development to the corresponding plot in Figure 36. The plot for the average pixel accuracy (red) shows the discussed accuracy gain for the patch sizes 3×3 through 9×9 and the following stagnation at around 83%.
While this chapter focused on the comparison with the results achieved in [25] and the influence of the choice of groundtruth on the evaluation process, the next chapter lays its emphasis on the influence of the chosen unary potentials.

9.3.2 Different Unary Potentials

This chapter examines the influence of the two different sets of unary potentials available for the MSRC-21 dataset on the semantic segmentation accuracy. Since the different sets of potentials were calculated by different methods, the split into training and test data also differs. For that reason a reduced test dataset was built from the intersection of the two respective test sets (see Chapter 7.2 for details).
Table 24 establishes the baseline for the following evaluation results on this reduced test set for the potentials calculated with the ALE and the state-of-the-art potentials respectively. The values included in the table capture the evaluation results for the MLE on the unary potentials and the results of the application of a single graph cuts regularization step on the same unary potentials.

Table 24: MSRC-21: Evaluation results for MLE on unary potentials and single graph cuts regularizations on potentials, left: potentials calculated with ALE, right: potentials as used in [25].

	ALE potentials			potentials as used in [25]		
	MLE	**GC:** $\alpha = 500$, $\beta = 10$	**GC:** $\alpha = 500$, $\beta = 50$	**MLE**	**GC:** $\alpha = 500$, $\beta = 10$	**GC:** $\alpha = 500$, $\beta = 50$
building	66.73	61.54	31.48	72.13	73.41	73.86
grass	81.26	83.79	81.16	88.54	88.63	89.93
tree	74.65	63.67	56.25	82.15	81.34	81.99
cow	91.59	86.24	27.21	87.71	88.80	89.74
sheep	71.31	18.60	0.00	91.81	92.77	93.62
sky	92.25	86.78	66.85	82.64	88.66	87.26
aeroplane	90.11	66.27	0.00	90.16	89.34	88.61
water	73.85	81.02	83.24	62.47	65.90	66.57
face	90.47	31.22	41.13	86.93	88.80	90.00
car	76.43	67.38	65.45	93.65	94.90	95.34
bicycle	88.84	92.21	97.70	90.36	90.42	89.93
flower	74.83	73.71	73.71	86.17	87.30	88.35
sign	61.28	62.70	43.94	76.31	87.98	94.69
bird	80.18	45.72	31.33	43.49	46.32	45.27
book	98.00	99.99	99.99	94.05	94.84	95.48
chair	68.88	58.99	23.82	56.06	58.15	57.89
road	76.58	60.70	48.33	87.68	89.24	89.00
cat	43.97	0.00	0.00	72.24	73.91	73.14
dog	69.78	46.64	46.72	41.55	35.73	35.61
body	82.96	50.91	23.69	73.95	79.19	78.71
boat	56.97	0.72	0.00	18.76	19.32	18.67
class accuracy	76.71	58.99	44.86	75.18	76.90	77.32
pixel accuracy	78.90	71.96	61.87	80.86	82.49	83.01

The comparison of the left and right columns in Table 24 shows that the various classes achieve different class accuracies for the respective evaluations of the MLE on the unary potentials. The classes 'building', 'grass', and 'tree' for example reach a class accuracy that is between 5 and 7 percent higher for the state-of-the-art potentials than for the ALE potentials. The classes 'cow', 'sky', or 'water' on the other hand achieve higher class accu-

racies for the ALE potentials. This results in an average class accuracy for the two different evaluations that only differs by 1.5% and an average pixel accuracy that differs by 2% for the evaluation on the MLE. In case of the pixel accuracy the higher results are achieved with the ALE potentials and for the class accuracy the state-of-the-art potentials yield the higher results. The application of the graph cuts approach on the ALE potentials shows a decline of the segmentation accuracy for both the class accuracy and the pixel accuracy. For the better state-of-the-art potentials the application of the graph cuts regularization leads to an improvement for both accuracies. For the state-of-the-art potentials all evaluation results are very similar to the corresponding results with the same parameters in Table 21.
Table 25 presents the segmentation accuracies achieved for patch sizes varying between 3×3 and 15×15 for the potentials calculated with the Automatic Labelling Environment.
The table shows a slight increase in accuracy for most of the classes for the smaller patch sizes of 3×3 through 9×9 or 11×11. This results in average pixel and class accuracies that also increase slightly until a patch size of 11×11 is reached in the case of the class accuracy and until patch size 13×13 for the pixel accuracy. Even though the improvement for the increasing patch sizes, in terms of segmentation accuracy, is not very pronounced in case of the MSRC-21 dataset and the ALE potentials, the accuracies exceed the baselines set by the MLE and the single graph cuts regularization on the unary potentials presented in Table 24 for each of the chosen patch sizes.
Figure 38 visualizes the average pixel accuracy and the average class accuracy presented in Table 25 for the weight parameters $\alpha = 500$ and $\beta = 50$ on the reduced test dataset.

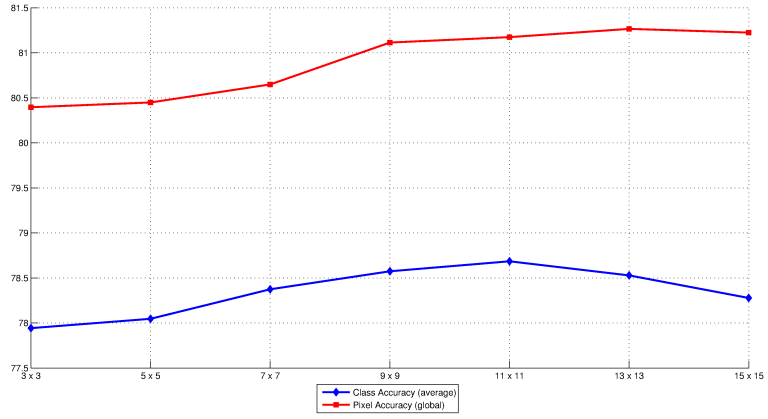

Figure 38: MSRC-21: Evaluation results for varying patch size on reduced dataset, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

Table 25: MSRC-21: Evaluation results for varying patch size on reduced dataset, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
building	67.71	67.96	67.98	67.82	67.77	67.57	67.17
grass	82.62	82.61	82.64	83.26	83.12	83.71	83.83
tree	74.35	74.21	74.56	74.44	74.28	73.62	73.16
cow	92.97	93.30	93.53	93.53	93.59	93.41	93.12
sheep	71.58	71.63	71.58	71.40	71.49	71.39	71.83
sky	96.32	96.38	96.44	96.25	96.32	96.64	96.49
aeroplane	91.27	91.38	91.72	91.83	92.09	91.12	89.60
water	78.94	79.07	79.86	83.07	83.90	84.22	84.96
face	89.40	90.07	89.77	89.70	89.80	89.80	89.43
car	77.85	77.55	77.36	77.17	77.20	77.15	77.17
bicycle	88.25	88.40	88.24	88.35	87.48	87.56	87.40
flower	75.64	75.71	75.73	76.38	76.79	76.45	75.84
sign	62.34	62.56	62.56	62.54	62.54	62.52	62.52
bird	86.59	86.90	86.82	86.94	86.77	86.78	86.45
book	98.33	98.30	98.37	98.33	98.34	98.32	98.32
chair	68.28	67.99	67.61	68.27	67.44	65.78	65.29
road	75.74	75.75	75.60	75.77	75.82	75.76	75.88
cat	43.15	42.83	49.24	48.80	50.58	51.30	51.21
dog	74.49	74.88	75.35	76.49	77.93	78.02	78.01
body	85.58	85.41	86.02	85.23	85.65	85.56	85.12
boat	55.41	56.10	54.89	54.46	53.47	52.39	51.03
class accuracy	77.94	78.05	78.37	78.57	**78.68**	78.53	78.28
pixel accuracy	80.40	80.45	80.65	81.11	**81.17**	81.26	81.22

The plot for the pixel accuracy (red) in Figure 38 shows the accuracy gain of 0.8% over the course of the increasing patch sizes from 3×3 through 11×11. The class accuracy (blue) also experiences an accuracy increase of 0.8% from patch size 3×3 until the patch size 9×9 is reached. At patch size 11×11 the pixel accuracy stagnates and at 9×9 the average class accuracy starts declining again.
Table 26 shows the results of the evaluation on the semantic segmentation results calculated on the same reduced test set as for the previous evaluation with the same weight parameters of $\alpha = 500$ and $\beta = 50$ but in this case with the qualitatively better state-of-the-art unary potentials as used in [25].

Table 26: MSRC-21: Evaluation results for varying patch size on reduced dataset, potentials as used in [25], $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
building	72.44	72.68	72.67	72.94	73.17	73.09	73.10
grass	89.87	89.96	90.27	90.65	90.88	91.08	91.31
tree	81.56	81.44	81.19	80.81	81.30	81.06	80.45
cow	88.15	88.53	88.14	88.31	88.04	87.78	87.69
sheep	91.56	91.60	91.45	90.90	90.79	90.60	92.26
sky	87.22	87.28	87.42	87.74	88.10	88.20	88.23
aeroplane	88.39	88.42	88.63	88.57	88.39	87.32	86.87
water	67.02	67.15	67.81	68.62	68.46	68.62	68.89
face	89.32	89.15	89.12	89.22	89.01	88.52	86.82
car	95.43	95.39	95.59	95.27	93.90	93.84	92.98
bicycle	89.20	89.39	89.76	89.25	88.50	88.12	87.27
flower	88.01	88.36	88.81	88.93	88.57	89.46	89.56
sign	91.45	91.62	93.55	93.61	97.15	97.15	97.11
bird	43.69	44.50	44.40	43.42	42.95	43.14	42.97
book	95.19	95.21	95.66	95.81	96.20	96.29	96.29
chair	57.19	57.00	57.09	57.92	58.81	58.20	57.63
road	88.96	88.68	88.89	89.31	89.28	88.85	88.81
cat	72.08	72.31	73.08	72.65	72.04	71.75	71.39
dog	39.55	39.62	40.88	41.00	40.34	38.89	38.54
body	73.28	78.13	78.04	75.30	75.25	75.44	75.14
boat	16.92	14.99	14.36	12.87	12.54	12.20	14.46
class accuracy	76.50	76.73	76.99	76.81	**76.84**	76.65	76.56
pixel accuracy	82.58	82.70	82.94	83.09	**83.21**	83.18	83.17

For the case of the segmentation results achieved with the state-of-the-art potentials the accuracy improvement over the course of the increasing patch sizes is even less pronounced. The individual classes show only very little improvement or decline in terms of class accuracy which results in a very small overall gain for the growing patch sizes in case of the average pixel accuracy of 0.6%. For the average class accuracy the values increase for the patch sizes between 3×3 and 7×7 and start to decline after that. Again the accuracy improvement with growing patch size is not very strong but the baselines established by the MLE and graph cuts on the unary potentials can be almost equaled or even exceeded for the larger patch sizes.

Figure 39 illustrates the average pixel accuracy and the average class accuracy for the weights of $\alpha = 500$ and $\beta = 50$.

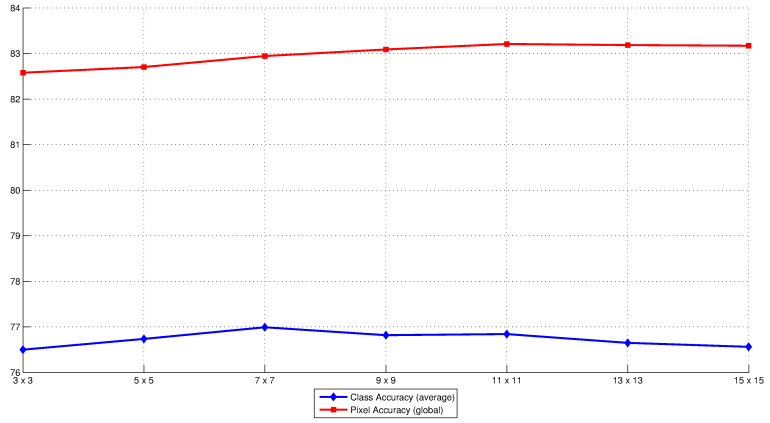

Figure 39: MSRC-21: Evaluation results for varying patch size on reduced dataset, potentials as used in [25], $\alpha = 500$, $\beta = 50$.

The average pixel accuracy (red) presented in Figure 39 shows a very slight increase from 82.6% for patch size 3×3 to 83.2% for patch size 11×11 after which the plot stagnates. The average class accuracy (blue) increases from patch size 3×3 from a value of 76.5% to 77% for patch size 7×7. For the larger patch sizes the average class accuracy declines again.

9.3.3 Confusion Matrix

The following table presents the evaluation of the best segmentation results in the form of a confusion matrix. The presented results were achieved with a patch size of 9×9 and weight parameters of $\alpha = 500$ and $\beta = 50$. The experiments were conducted on the same test set that was used in [25] and the evaluation was performed on the relabeled groundtruth.
Table 27 presents the percentage of correctly classified pixels per class in the diagonal of the matrix. The values are identical to those presented in Table 23. The classes 'grass', 'cow', 'sky', 'face', 'car', 'bicycle', 'flower', and 'book' achieve very high class accuracies of around 90%. The classes 'building', 'tree', 'sheep', 'aeroplane', 'water', 'sign', 'road', 'cat', and 'body' reach an accuracy between 70% and 80%. The classes 'bird', 'chair', and 'dog' still achieve a class accuracy of around 50% and only the class 'boat' that is underrepresented in the test images has a class accuracy of only 20%. Most of the classes are most frequently confused with the classes they are usually surrounded by, as for example 'cow' and 'sheep' with 'grass' or 'boat' with 'water'.

Table 27: MSRC-21: Confusion matrix for patch size 9×9, $\alpha = 500$, $\beta = 50$.

	building	**grass**	**tree**	**cow**	**sheep**	**sky**	**aeroplane**	**water**	**face**	**car**	**bicycle**	**flower**	**sign**	**bird**	**book**	**chair**	**road**	**cat**	**dog**	**body**	**boat**
building	**72.84**	3.56	4.20	0.05	0.01	1.39	0.25	0.48	1.52	2.71	1.97		0.29	0.37	4.50	0.94	3.58		0.16	0.78	0.40
grass	0.09	**90.81**	2.24	0.77	0.70	0.02	0.37	0.02	0.02	0.00	0.12	1.82		0.12	0.00	0.77	1.57	0.00	0.08	0.48	
tree	2.17	3.99	**81.85**	0.02		3.14	0.89	1.30	0.52	0.14	0.24	3.54	0.64	0.14	0.17	0.09	0.33			0.83	
cow	0.05	7.96	0.12	**88.54**	0.30			0.07	0.55	0.08		0.51						1.13	0.42	0.28	
sheep	3.70	8.44	0.58	2.52	**83.26**												1.11		0.38		
sky	4.71	0.16	1.63			**91.01**	0.14	2.19	0.00	0.00			0.03	0.00	0.06		0.00		0.00	0.02	0.02
aeroplane	8.54	5.09	0.49			1.34	**84.26**										0.29				
water	2.04	7.87	2.60	0.01		1.05	0.03	**66.81**	0.02	3.77	0.05			5.60	0.08	0.09	8.95		0.26	0.49	0.29
face	2.55	0.07	1.10	0.20		0.49		0.15	**87.25**	0.06		0.08	0.01		0.48		0.03		0.04	7.48	
car	2.79		0.17			0.24				**87.24**	1.41			0.46	0.00		7.62		0.07		
bicycle	3.06	0.01	0.38						0.01	0.77	**90.40**				0.63	0.13	4.60				
flower		0.09	5.49					0.52			0.29	**93.22**		0.34					0.04		
sign	16.83		0.34	0.08		0.38						0.00	**82.07**		0.24		0.06				
bird	1.66	6.99	3.58	1.97	1.28	0.54		13.59						**47.00**		4.58	3.70	9.76	5.00	0.12	0.23
book	2.86								0.15			1.18			**95.10**					0.71	
chair	10.73	9.33	1.51	5.82		0.56					4.52			1.06		**62.11**	4.11			0.27	
road	1.86	1.89	0.17	0.07	0.42	0.47	0.63	1.99	0.00	2.61	1.09			0.13	0.01	0.56	**85.34**	0.98	0.70	1.08	
cat	0.14		1.17					1.62		3.53	0.86			0.60			12.33	**79.74**			
dog	5.09	1.77	1.53	6.62	4.54	0.31	0.11	0.18	8.32		0.16			0.24	0.00	0.61	18.21	0.10	**51.62**	0.59	
body	1.35	3.63	1.14	0.54	2.26	0.04		0.18	4.88	0.63		0.76	1.34		0.89		2.47		1.09	**78.80**	
boat	23.92	0.03				0.48	1.11	28.75		8.19	13.01			1.99	0.00	0.88	2.19				**19.46**

9.3.4 Example Segmentation Results

This chapter presents some examples of the semantic segmentation results that the implemented system could achieve on the MSRC-21 dataset. Figures 40 and 41 show eight examples with the input image at the top row, followed by the annotated groundtruth in the second row, and the semantic segmentation results calculated with the ALE potentials and the state-of-the-art potentials in the third and fourth row respectively. The presented groundtruth is the relabeled one and the parameters that were used in the segmentation process are a patch size of 11×11 and weight parameters of $\alpha = 500$ and $\beta = 50$.

Figure 40: MSRC-21: Example images, corresponding groundtruth, and segmentation results with ALE potentials and potentials as used in [25] (top to bottom, respectively) for patch size 11×11, $\alpha = 500$, and $\beta = 50$ (Part 1).

The example segmentation results in Figures 40 and 41 show that the results can achieve a high level of similarity to the annotated groundtruth for most of the input images. In some cases, like the first three examples in Figure 40, the system yields almost identical outputs for the different unary potentials. For other images either the ALE potentials or the state-of-the-art potentials can lead to the more accurate segmentation results. In some cases, like the last two examples in Figure 41, the unary potentials used as input are flawed and the system yields results that are obviously not correct.

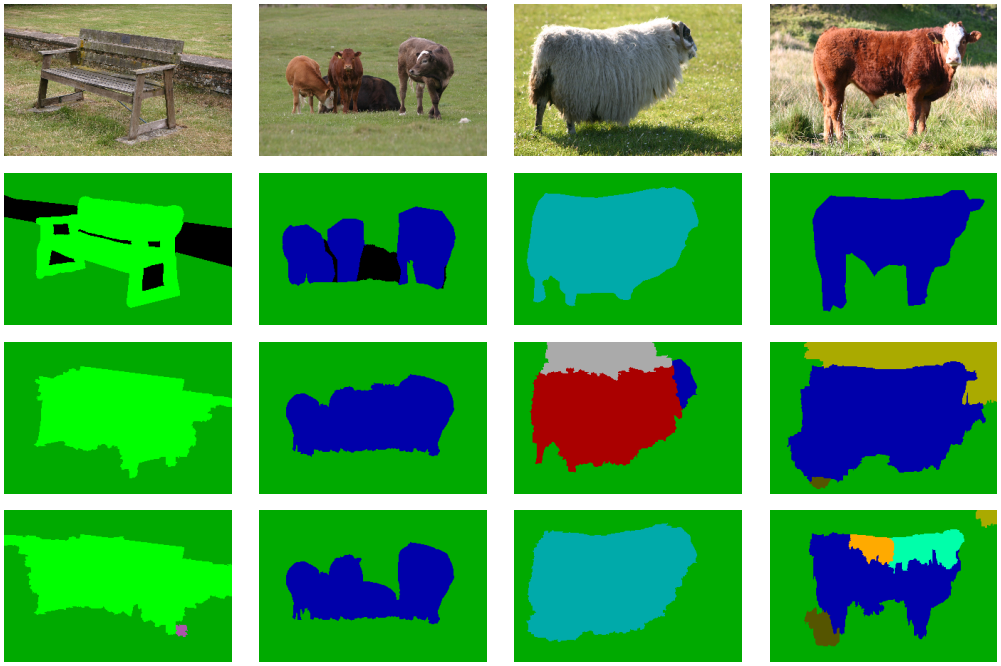

Figure 41: MSRC-21: Example images, corresponding groundtruth, and segmentation results with ALE potentials and potentials as used in [25] (top to bottom, respectively) for patch size 11×11, $\alpha = 500$, and $\beta = 50$ (Part 2).

9.3.5 Summary

The MSRC-21 dataset with its two available sets of unary potentials and its two different sets of groundtruth data offers an excellent opportunity to examine the influence of the deployment of different unary potentials in the implemented method and the impact of different groundtruth data in the evaluation process.

Table 28: MSRC-21: Comparison of achieved results with state-of-the-art.

Method	**accuracy**	
	class	**pixel**
Shotton et al. [42]	-	72.2%
Krähenbühl & Koltun [25]	86.0%	78.3%
Ladický et al. [27]	87%	77%
Method (groundtruth as used in [25])	84.65%	76.94%
Method (relabeled groundtruth)	83.05%	77.08%

Table 28 shows a comparison of the segmentation accuracies achieved with the most successful set of parameters for the implemented method with the accuracies reported by the state-of-the-art methods. The semantic segmentation accuracies reported in papers discussed in Section 2 range between 72.2% and 78.3% for the average pixel accuracy and 86% and 87% for the average class accuracy. The implemented method can achieve similar results on the same test dataset for the class accuracy and also comes very close for the pixel accuracy. The evaluation also shows that the presented method can reach an improvement in terms of accuracy by increasing the chosen patch size for the patch prior. The example segmentation results presented in the previous chapter show that a high level of similarity between the output of the system and the groundtruth can be reached for many of the input images.

9.4 Sowerby

This chapter presents the semantic segmentation results that can be achieved with the implemented semantic segmentation framework on the Sowerby dataset presented in Chapter 7.3. The outcomes are presented in the form of tables and diagrams along with a discussion of the results.

9.4.1 Segmentation Accuracy and Patch Size

Table 29 shows the three baseline segmentation accuracies similar to the two previous chapters. The weight parameters were again chosen with values of 500 for α and 10 and 50 for the two different choices of β respectively. The potentials used in the following evaluations are again calculated with the Automatic Labelling Environment.

Table 29: Sowerby: Segmentation accuracies for MLE and single graph cuts regularizations on unary potentials, potentials calculated with ALE.

	MLE	**Graph Cuts** $\alpha = 500, \beta = 10$	**Graph Cuts** $\alpha = 500, \beta = 50$
sky	89.10	94.04	91.31
vegetation	76.61	79.31	75.01
road marking	1.03		
road surface	87.86	79.45	69.94
building	38.61	30.47	18.30
street objects	1.81	1.73	
car	18.55	10.25	5.42
class accuracy	44.79	42.18	37.14
pixel accuracy	80.11	78.29	71.85

Table 29 once again shows the limitations of the potentials calculated with the ALE. The graph cuts regularization on the unary potentials again leads to a decline of the segmentation accuracy compared to the simple MLE on the unary potentials. For most of the classes the segmentation accuracy is lower, compared to the MLE on the unary potentials, after a single graph cuts regularization step except for the class 'sky' that can achieve a higher accuracy for both sets of weight parameters, and 'vegetation' that can reach a 3% higher accuracy for $\beta = 10$ than the MLE. Especially the classes 'road marking' that has already a very low value of only 1% and 'street objects' (2%) get completely eliminated by the graph cuts regularization. These low values are based on the small size of the images and the even smaller size of

the corresponding structures in these images. These low values in turn lead to a very low average class accuracy.
Table 30 shows the first set of segmentation results for weight parameters $\alpha = 500$ and $\beta = 10$ that should in the best case exceed the baselines presented in the previous table.

Table 30: Sowerby: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

	3 x 3	**5 x 5**	**7 x 7**	**9 x 9**	**11x11**	**13x13**	**15x15**
sky	94.32	94.60	94.75	95.14	95.07	93.90	92.33
vegetation	82.79	81.56	81.83	82.63	82.25	81.60	77.42
road marking	1.18						
road surface	92.07	91.91	91.90	91.28	90.75	90.43	90.08
building	35.94	36.42	35.77	33.12	29.20	24.15	11.37
street objects	1.73	1.73	1.73	1.73			
car	16.53	11.61	4.83				
class accuracy	**46.37**	45.41	44.40	43.41	42.47	41.44	38.74
pixel accuracy	**84.91**	84.37	84.45	84.45	83.88	83.08	80.35

Table 30 shows that the segmentation accuracy decreases over the course of the rising patch size for most of the semantic classes. Only the class 'sky' shows a slight accuracy increase for increasing patch size until the larger sizes of 13×13 and 15×15 are reached. This results in a gradual decrease of both the average class accuracy and the average pixel accuracy. This behavior results from the small images that the Sowerby dataset consists of. For the smaller patch sizes of 3×3 and 5×5 the average pixel and class accuracy results exceed the baseline results established by the MLE and graph cuts regularizations on the unary potentials from Table 29. While the average pixel accuracy still stays well above the baseline for the patch sizes of 7×7 through 13×13, the average class accuracy strongly declines for these patch sizes. These results give the first indication that the developed method is less suitable for images below a certain size. Especially smaller objects, like in the case of the Sowerby dataset objects consisting of pixels of the classes 'road marking', 'car', and 'street objects', that are already severely underrepresented by the unary potentials, get completely eliminated for the larger patch sizes.
Figure 42 again visualizes the average class and average pixel accuracies in the form of a diagram.

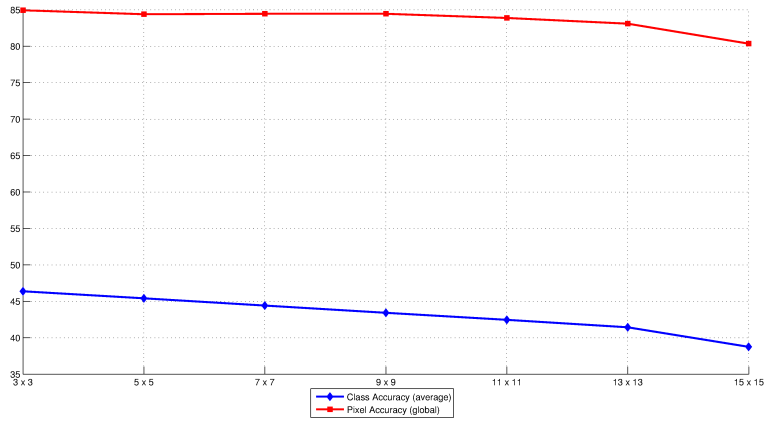

Figure 42: Sowerby: Evaluation results for varying patch size, potentials calculated with ALE, $\alpha = 500$, $\beta = 50$.

The plots illustrate what has already been discussed in the previous paragraph. The plot for the pixel accuracy (red) starts out at an accuracy of 85% for the patch sizes of 3×3 and slowly declines to 83% for size 13×13 until drastically dropping to 80% for 15×15. The class accuracy (blue) shows a much faster decline from 46.4% for patch size 3×3 down to 38.7% for the maximum image patch size of 15×15.
Other choices for the weight parameters α and β yield similarly bad results and are for that reason not included in the further discussion.

9.4.2 Confusion Matrix

This chapter presents the results for the best outcome of the experiments on the Sowerby dataset in the form of a confusion matrix. The parameter set that yields the best results consists of $\alpha = 500$ and $\beta = 50$. As discussed earlier, the small image size causes the implemented method to achieve worse results for the larger patch sizes than for the small patches. While there is still a slight accuracy increase compared to the baseline set by the MLE on the unary potentials for the larger patch sizes, the best results in terms of accuracy can be achieved with the minimum patch size of 3×3 for this dataset.
The diagonal of the confusion matrix holds the per class accuracy values from Table 30. The highest accuracies of 94.32% and 92.07% can be achieved for the classes 'sky' and 'road surface' respectively. Both classes show their highest confusion index of 4.4% and 7.78% with class 'vegetation' respectively. Class 'vegetation' reaches with 82.79% the next highest accuracy and is most frequently confused with 'sky', 'road surface', and 'building' instances. The

Table 31: Sowerby: Confusion matrix for patch size 3×3, $\alpha = 500$, $\beta = 50$.

	sky	**vegetation**	**road marking**	**road surface**	**building**	**street objects**	**car**
sky	**94.32**	4.40		0.07	1.21	0.00	
vegetation	7.26	**82.79**		5.40	4.29	0.02	0.23
road marking	0.30	0.59	**1.18**	94.83	3.10		
road surface	0.00	7.78	0.02	**92.07**	0.11		0.01
building	4.28	53.00		4.72	**35.94**	1.06	1.00
street objects	14.92	51.85		12.58	16.80	**1.73**	2.11
car	0.51	28.31		22.97	31.69		**16.53**

class 'building' shows with only 36% a poor performance and the class 'vegetation' that it is easiest confused with achieves a confusion index of 53% for this class. Class 'car' performs even worse and only reaches an accuracy of 16.5%. It is mostly confused with the classes 'building', 'vegetation', and 'road surface', by which it is most frequently surrounded. Finally, the classes 'street objects' and 'road marking', that only form very small structures in the small images, are almost nonexistent in terms of segmentation accuracy. The 'street objects' class is in 53% of all cases confused with the 'vegetation' class and also in almost equal parts by 'building', 'sky', and 'road surface'. The 'road marking' class is almost exclusively confused with the 'road surface' class that it appears on in the images.

9.4.3 Example Segmentation Results

This chapter concludes the discussion of the evaluation for the Sowerby dataset and shows some examples of the semantic segmentation results that could be achieved with the presented method. Figure 43 shows four examples with the original image at the top row, the desired result consisting of the labeled groundtruth in the middle row, and the segmentation results that were calculated with the implemented method for the parameter set of $\alpha = 500$, $\beta = 50$, and a patch size of 3×3 at the bottom.
The results presented in Figure 43 show the problem that comes with the small size of the images in the Sowerby dataset. Very small structures like the road markings or the road sign in the rightmost image get eliminated by the

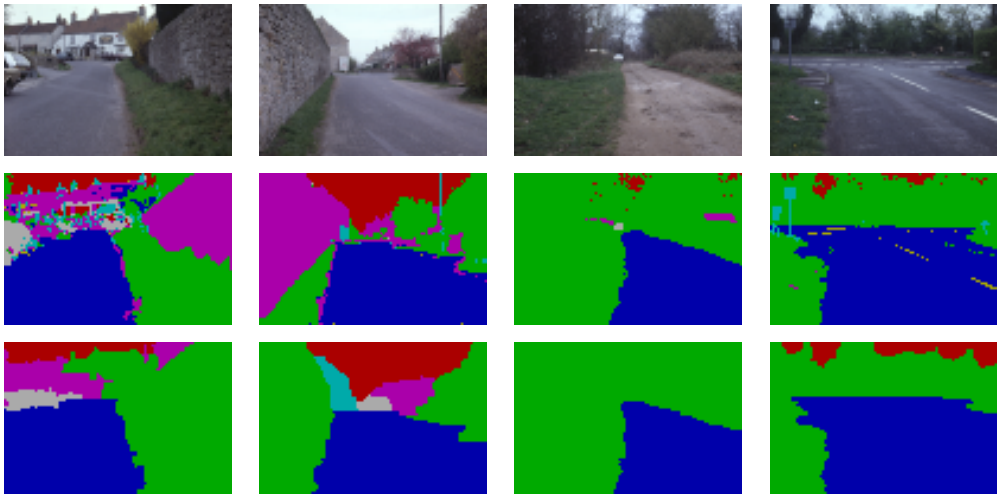

Figure 43: Sowerby: Example images, corresponding groundtruth, and achieved segmentation results (top to bottom, respectively) for patch size 3×3, $\alpha = 500$, $\beta = 50$.

graph cuts regularization. This is also true for example for the small areas of pixels labeled as 'sky' (red) in the second image from the right. On the other hand, similar areas, that are a bit larger, in the rightmost image, get enlarged over the course of the iterations. The overall similarity between the segmentation results and the desired outcome represented by the annotated groundtruth in the middle row is fairly high.

9.4.4 Summary

The evaluation of the results in the previous chapter shows the limitations for the implemented method when dealing with images below a certain size. The very small structures in the original images lead to very small structures in the calculated unary potentials, if they are captured at all. The iterative modification of the unary potentials and the subsequent application of the graph cuts approach frequently lead to a complete elimination of these tiny structures over the course of the iterations. The result is a very low segmentation accuracy for the corresponding class (sometimes even 0), which in turn leads to poor average class accuracies.
Table 32 shows a comparison of the segmentation accuracies reported by the state-of-the-art methods with the best results achieved with the implemented method.
While the achieved average pixel accuracy of almost 85% is quite good compared to the one achieved for the Corel-100 dataset (74.66%), the class ac-

Table 32: Sowerby: Comparison of achieved results with state-of-the-art.

Method	**accuracy**	
	class	**pixel**
Shotton et al. [42]	-	88.6%
He et al. [19]	-	89.5%
Toyoda & Hasegawa [46]	-	90.0%
Method	46.37%	84.91%

curacy of only 46.37% is pretty low. This low value is based on the fact that the very small structures of the classes 'road markings' and 'street objects', that get almost eliminated with the graph cuts approach, lower the average value of the class accuracy. Again the reported state-of-the-art results ranging from 88.6% to 90.0% can not be exceeded, but the established baselines of 80% and lower from Table 29 can always be topped with the implemented method. The example segmentation results presented in the previous chapter show that a high similarity between the desired output, represented by the groundtruth, and the segmentation results, achieved with the implemented method, can be reached. The analysis of the results in this chapter shows that while larger patch sizes also lead to still decent results compared to the baseline of applying the MLE to the unary potentials, the best results are achieved for the minimum patch size of 3×3. This leads to the conclusion that the presented method is not very suitable for very small images.

This section presented the experimental results that could be achieved with the semantic segmentation framework implemented in the course of this master's thesis. First the means of evaluations were discussed with an introduction of the measures for the segmentation accuracy and the evaluation framework as well as a short discussion of the different experiments that were conducted. Then the experimental results for each of the three datasets presented in Section 7 followed. The outcomes of these experiments were presented in the form of tables and diagrams with corresponding interpretations of the results. The results show that the implemented method can achieve good results that are comparable to the state-of-the-art results although the presented method follows a much simpler approach than most of the other methods. The implemented system can achieve the task that it set out to fulfill, which is a gradual improvement of the segmentation results with the application of an iterative approach. Another fact that was made obvious by the experiments is that the presented method strongly depends on the

quality of the precomputed unary potentials. If they are flawed, the system can not compensate for that circumstance, and if they already yield sufficiently accurate segmentations, the system only leads to very small further improvements.
The next section completes this master's thesis with the conclusion. The achieved results are discussed one final time along with the strengths and weaknesses of the implemented method. Finally, a short outlook is given for possible future improvements to the presented method.

10 Conclusion

This section presents a final discussion of the results that could be achieved with the implemented semantic segmentation framework. The system is based on the current state-of-the-art method of interpreting the semantic segmentation problem as an energy optimization problem and like most of the other state-of-the-art approaches it uses graph cuts regularization as inference method. The implemented system enhances the standard semantic segmentation pipeline by introducing prior knowledge on the semantic level with the help of databases that are generated from label patches of different sizes. The prior knowledge consists of a histogram of occurrences for all label patches that are extracted from a set of training images. The resulting databases are then used to determine the probability for each possible class at every pixel location under the learned prior for the intermediate segmentation results, and to update them accordingly. This process is repeated iteratively until convergence. The goal of this method is to avoid configurations of semantic categories that are unlikely or impossible under the learned prior. Even though the approach used by the implemented method is much simpler than most used by other methods presented in Section 'Related Work', comparable results can be obtained, as presented in the previous section.
The experiments show that the semantic segmentation results achieved with the implemented method in general have a high level of similarity with the desired segmentation, represented by the annotated groundtruth. The tests also proved that the system meets one of the goals that was set prior to the start of the implementation, which was to gradually improve the semantic segmentation based on a rough initial segmentation by using an iterative approach. The method actually does show a step-by-step improvement of the semantic segmentation and a fast convergence in only a low one-figure number of steps for most of the input images.
The main improvement in the segmentation accuracy can be reached at the borders between two different class labels. This generally results in a smoothing of the edges between objects of two semantic classes, as the presented exemplary segmentation results can show. As the experiments with the MSRC-21 dataset in Chapter 9.3 showed, this can lead to minor problems in the course of the evaluation process if the annotated groundtruth is not defined very well in those regions. This is one of the weaknesses of the presented method. It demands the use of a sufficiently annotated groundtruth in the training phase to obtain an adequate number of fully labeled label patches from the training images. If the training data contains many unlabeled pixels, the system learns label patches including the 'void' class that will not occur in the intermediate segmentation results.

Another fact, that was especially manifested in the experiments with the MSRC-21 dataset, was the influence of the quality of the precomputed unary potentials. If those potentials already yield a sufficiently accurate semantic segmentation, as is the case for the publicly available set of potentials for the MSRC-21 dataset, the presented system can only yield a negligibly small further improvement in terms of segmentation accuracy. If on the other hand the potentials are imprecise and not particularly smoothed at the transitions between labels, as is the case for the potentials calculated with the Automatic Labelling Environment, the segmentation accuracy improvements that can be achieved with the implemented method can be very high. One thing that the presented framework can not compensate for, is the occurrence of flawed potentials. If the precomputed unary potentials contain large areas of pixels that are labeled incorrectly, the system is unable to eliminate them. These facts lead to the conclusion that the implemented system strongly depends on the quality of the precalculated unary potentials.
Another goal of this master's thesis was to examine the influence of the chosen patch size on the quality of the semantic segmentation results. The experiments prove that a prior learned on patches of larger size can lead to more accurate segmentation results in many cases. This circumstance is based on the fact that larger label patch sizes lead to priors that are much more precise. The results show that a gradual improvement of the segmentation accuracy can be achieved in most cases up until a certain point is reached where the accuracy starts to decline again. These tests also showed that the method is only suitable for images above a certain size. The results for the Sowerby dataset in Chapter 9.4 show, that for images that are too small the learned prior and the resulting regularizations lead to a decrease in segmentation accuracy for larger patch sizes because the tiny details included in the small images are eliminated in most cases.
One last weakness that the presented semantic segmentation framework experiences is the fact that it is not capable of calculating the segmentation results in a real-time appropriate time frame since the precomputation of the unary potentials and the precomputation of the patch prior database in the training phase is very time consuming.

While the accuracy of the semantic segmentation results achieved with the presented approach is sufficiently high, considering the relative simplicity of the method, there is still possible room for improvement.
One possibility for further improvement of the system is to replace the simple approach of using a histogram of label patch occurrences by using a Gaussian Mixture Model (GMM) or Mixture of Multinomials.
Another approach would be to take similarities between label patches into

account. The current implementation only considers the number of occurrences of learned label patches that have the exact same configuration as the label patch under examination. If only one pixel differs, the corresponding label patch is not considered in the calculation of the prior. This shortcoming could be circumvented by encoding similarities between label patches in the prior database with the help of some kind of median patch for similar class label constellations. This would also result in a reduction of the necessary training images to generate a sufficiently large prior database.

A Abbreviations

ALE	Automatic Labelling Environment
CRF	Conditional Random Field
EPLL	Expected Patch Log Likelihood
GC	Graph Cuts
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HoG	Histogram of Gradients
HOG	Histogram of Oriented Gradients
ICM	Iterated Conditional Modes
LBP	Loopy Belief Propagation
LUV	Luminance, Chrominances U and V (color model)
MAP	Maximum A Posteriori
mCRF	multiscale Conditional Random Field
MEMM	Maximum Entropy Markov Model
MLE	Maximum Likelihood Estimation
MRF	Markov Random Field
NN	Nearest Neighbor
SIFT	Scale Invariant Feature Transform

References

[1] R. J. Adler and J. E. Taylor. *Random Fields and Geometry*. Springer, 2007. 20

[2] K. Aneja, F. Laguzet, L. Lacassagne, and A. Merigot. Video-rate image segmentation by means of region splitting and merging. In *Proceedings of the IEEE International Conference on Signal and Image Processing Applications (ICISPA)*, 2009. 15

[3] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik. Semantic segmentation using regions and parts. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 15

[4] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2011. 15

[5] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society (JRSS)*, 1986. 21

[6] S. Beucher. Watersheds of functions and picture segmentation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1982. 15

[7] S. Beucher and C. Lantuejoul. Use of watersheds in contour detection. In *International Workshop on Image Processing (IWIP)*, 1979. 15

[8] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1999. 22

[9] Y. Boykov, O. Veksler, and R. Zabih. A new algorithm for energy minimization with discontinuities. In *Proceedings of the International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 1999. 22

[10] P. C. Chen and T. Pavlidis. Image segmentation as an estimation problem. In *Proceedings of the IEEE Conference on Decision and Control including the Symposium on Adaptive Processes (CDC)*, 1979. 15

[11] D. Collins, W. Wright, and P. Greenway. The sowerby image database. In *Proceedings of the IEEE International Conference on Signal and Image Processing Applications (ICISPA)*, 1999. 48

[12] P. F. Felzenszwalb and R. Zabih. Dynamic programming and graph algorithms in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2011. 17, 22, 25, 31

[13] P. A. Ferrari, A. Frigessi, and P. G. de Sá. Fast approximate maximum a posteriori restoration of multicolour images. *Journal of the Royal Statistical Society (JRSS)*, 1995. 22

[14] P. A. Ferrari, M. D. Gubitoso, and E. J. Neves. Reconstruction of gray-scale images. *Methodology and Computing in Applied Probability (MCAP)*, 1997. 22

[15] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962. 30

[16] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1984. 25

[17] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society (JRSS)*, 1989. 21

[18] M. Haque, M. Murshed, and M. Paul. Improved gaussian mixtures for robust object detection by adaptive multi-background generation. In *Proccedings of the International Conference on Pattern Recognition (ICPR)*, 2008. 16

[19] X. He, R. S. Zemel, and M. A. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 18, 21, 45, 46, 48, 77, 98

[20] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. A Bradford Book, 1992. 25

[21] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 1983. 25

[22] S. Kluckner, T. Mauthner, P. M. Roth, and H. Bischof. Semantic classification in aerial imagery by integrating appearance and height information. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2009. 16, 18

[23] P. Kohli, L. Ladický, and P. H. Torr. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision (IJCV)*, 2009. 23

[24] P. Kontschieder, S. R. Bulo, H. Bischof, and M. Pelillo. Structured class-labels in random forests for semantic image labelling. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011. 23

[25] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in Neural Information Processing Systems (NIPS)*, 2011. 3, 4, 5, 6, 17, 27, 46, 47, 78, 79, 80, 81, 82, 83, 84, 86, 87, 88, 90, 91

[26] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005. 21

[27] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr. Graph cut based inference with co-occurrence statistics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010. 17, 27, 47, 91

[28] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2001. 21

[29] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision (IJCV)*, 2001. 16

[30] W. Li. Multi-threshold color image segmentation based on region growing. In *Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, 2010. 15

[31] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2011. 19

[32] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1999. 16

[33] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 2004. 16

[34] T. H. Morrin. A black-white representation of a gray-scale picture. *IEEE Transactions on Computers (TC)*, 1974. 15

[35] R. Ohlander, K. Price, and D. R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing (CGIP)*, 1978. 15

[36] N. B. Rais, M. S. Hanif, and I. A. Taj. Adaptive thresholding technique for document image analysis. In *Proceedings of the International Multitopic Conference (INMIC)*, 2004. 15

[37] E. A. Rashed, Z. Wang, and H. Kudo. Adaptive thresholding for robust iterative image reconstruction from limited views projection data. In *IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, 2011. 15

[38] T. Saikumar, M. Nagarani, K. Yojana, and B. Shashidhar. Image segmentation of an adaptive threshold algorithm using watershed transform and fuzzy c-means clustering on level set method. In *Proceedings of the International Conference on Advances in Engineering, Science and Management (ICAESM)*, 2012. 15

[39] A. Satpathy, X. Jiang, and H.-L. Eng. Extended histogram of gradients feature for human detection. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2010. 16

[40] Y. Schnitman, Y. Caspi, D. Cohen-Or, and D. Lischinski. Inducing semantic segmentation from an example. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2006. 19

[41] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006. 16, 46, 52

[42] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision (IJCV)*, 2009. 16, 18, 27, 45, 46, 47, 48, 77, 80, 91, 98

[43] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. Thomson-Engineering, 2007. 15

[44] G. Stockman and L. G. Shapiro. *Computer Vision*. Prentice Hall PTR, 2001. 15

[45] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006. 20, 21

[46] T. Toyoda and O. Hasegawa. Random field model for integration of local information and global information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2008. 20, 45, 46, 48, 77, 98

[47] V. Černý. A thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications (JOTA)*, 1985. 25

[48] O. Veksler. *Efficient Graph-Based Energy Minimization Methods in Computer Vision*. PhD thesis, Cornell University, Ithaca, NY, USA, 1999. 17, 22, 25, 33

[49] A. Vezhnevets, V. Ferrari, and J. Buhmann. Weakly supervised structured output learning for semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 16

[50] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005. 16

[51] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 1993. 22

[52] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems (NIPS)*, 2000. 21

[53] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011. 23, 35