ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

BIWI Computer Vision

Deformable shape models for 2D object segmentation

Semester Thesis

Robin Thandiackal
Department of Mechanical Engineering

Advisor: Dr. Mukta Prasad
Supervisor: Prof. Vittorio Ferrari

August 9, 2011

Abstract

Given a set of images showing individual 2D instances of an object class, the goal is to learn object class deformation in 2D for segmentation automatically. Class deformation is modelled by linear combinations of basis shapes. Usually, given segmentation data and correspondences, such basis shapes can be easily learned with Principal Component Analysis. Here, we are dealing with unsegmented RGB images. We show how to learn segmentations and deformation sequentially in an iterative framework. Variations of the basic algorithm are explained, tested and compared. In order to introduce smoothness priors and data dependent pairwise terms, Graph-cut can be incorporated. The final results show that explicitly restricting segmentations by a linear subspace of shape deformation, leads to significant improvements.

Acknowledgements

Special thanks go to my supervisor Mukta Prasad. Besides her professionalism and enthusiasm, she was at any time willing to help and supported me with her technical know-how, experience and creative ideas. Many thanks also to Prof. Ferrari with his great propositions, ideas and helpful constructive criticism.

Contents

Chapter 1

Introduction

1.1 Problem description

In our problem we are given a collection of images, where in each of them exactly one object instance of a class is represented. Based on this, the task is to compute the binary segmentation into object foreground and background in all the images. The solution shall be determined automatically and completely unsupervised.

The main contribution of this work is to describe the class objects with a learned *global deformable shape model*, and to use this to improve the process of segmentation. We aim for a linear basis model which is learned with Principal Component Analysis (PCA). Therefore, we set up an iterative algorithm framework for segmentation, where we can embed the model. The quality of the deformation model can finally be evaluated through the accuracy of the segmentations.

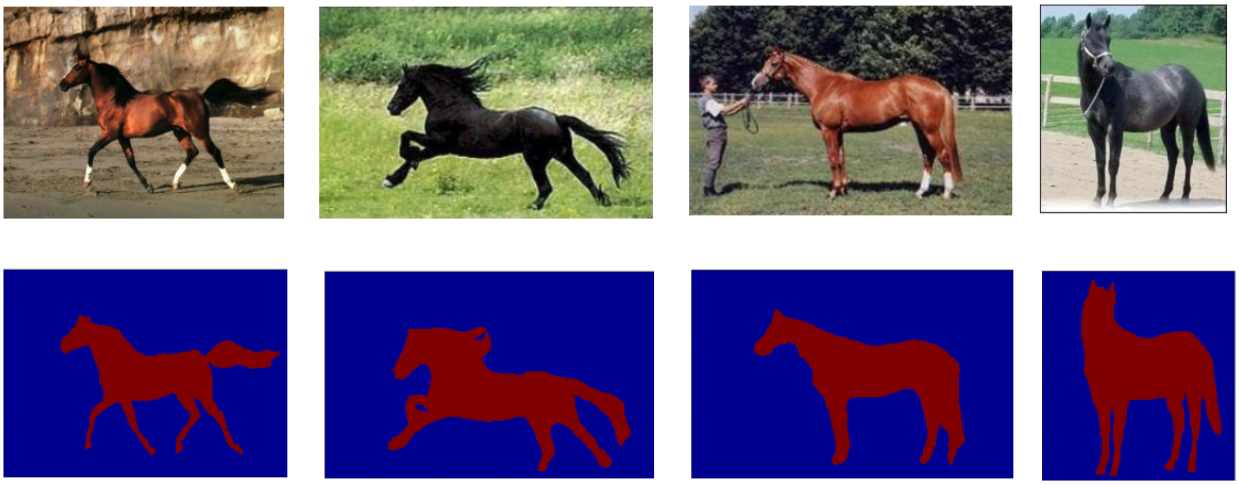

Figure 1.1: Examples from the horse dataset with corresponding ground-truth segmentations

1.2 Motivation and Challenges

One has to realise that although we are aiming for segmentations, the main focus is pointed to the learning of a meaningful deformable 2D shape model. Nevertheless, these processes are tightly coupled. The idea

of learning such a model is mainly motivated by the fact that we can explicitly describe and explain 2D object shape variations occurring in a certain class by means of a parameter set. In other words one could say that we are able to quantify the variations, while doing an abstraction. Often this process includes a compressional aspect when with a small amount of parameters still a great generalisation can be attained. Furthermore, these models can be advantageous to support tasks like segmentation, detection or recognition, because they restrict the space of possible solutions. Referring to the precise task of segmentation this would mean that the final segmented images would be encouraged to take not any arbitrary shape but one so that they don't conflict with the model.

Besides learning a deformation model from a set of images, we also aim to obtain this automatically. Of course one could think of supporting the learning process by means of human interaction, e.g. one could manually annotate certain 2D object boundaries. Nevertheless, especially if we are dealing with lots of images, this becomes time-consuming and very inefficient. In addition, when we want to implement such a segmentation framework in robot applications, clearly automatic techniques are required.

Whenever we deal with the matter of learning, which here regards the deformation model, there exist various possibilities to do so. Basically we could imagine supervised, reinforcement or unsupervised learning settings. The former characterises a setting in which one learns from a 'teacher' that provides the correct answers during a training phase. The reinforcement setting describes the learning motivated by rewards and the unsupervised learning aims to find common structures and similarities. Most suitable for our work is the latter as we expect good generalisation properties to arbitrary object classes without using any training at the same time.

What remains are challenges that we have to face in order to solve the given problem. It is necessary to derive a deformable 2D shape model which can describe all the object variations, while still restricting the shapes to be typical in the given class. Therefore, it is essential to explain the appearance of class objects by means of adequate features like colour or contours. Moreover it is crucial to intelligently embed the deformation model within the algorithm for segmentation. The model must be able to successfully restrict the individual segmentations to take their shape within the object class.

1.3 Thesis Organisation

Chapter 2 explains the basic algorithm. It is shown how the deformation model is embedded into an iterative algorithm for segmentation. Variations are explained as well.

Chapter 3 presents several experiments on an explicit test dataset. At first, the variations of the algorithm are evaluated with respect to ground-truth segmentations. Furthermore qualitative as well as quantitative results are explained.

Chapter 4 discusses and interprets the results of the latter chapter. Also propositions for potential improvements are given. Finally a conclusion regarding the use of deformable shape models for segmentation will be drawn.

Chapter 2

Algorithm

2.1 Existing segmentation framework

As our problem setting is the same as in ClassCut [1], it builds an ideal reference for our work. The main idea in ClassCut is to use the general framework of segmentation where an image is represented in terms of a graph. A graph-node represents a pixel with its label. The label describes whether the pixel belongs to foreground or background. Therefore, a segmentation is defined through a set of labels. The general framework says that we can further define conditions for the pixel labels such as smoothness or others. These can then be formulated in terms of energies of single pixels, called unary potentials or energies between neighbouring pixels, called pairwise potentials. Collecting all conditions we can derive a total energy of the graph. By minimising this energy one can find the optimal labels that fulfill the conditions best.

The major contribution of ClassCut is that unary and pairwise potentials are defined over the whole set of images, whereas in standard segmentation problems these terms are defined only for single images. Furthermore, the energy potentials are updated in an alternating process between learning an abstract class model and segmenting all the images. Keeping in mind all this, we have to notice that ClassCut differs fundamentally from the work presented here. Whereas we do, ClassCut does not use any explicit global deformation model to restrict the segmentations to be class-like.

2.2 Concept

Let us recall: Our task is to learn a global deformable shape model from the given image collection in order to segment each of the images into object foreground and background. Therefore, it is necessary to explain the deformation model that we are about to use.

We assume that, given a collection of abstract shapes s_j from a certain class, they lie in a subspace of representative basis shapes b_i. Using this, we can define a model for the class, or a deformable shape model how we call it.

$$s_j = \sum_{i=1}^{K} \alpha_{j,i} \cdot b_i, \;\; j = 1, ..., N, \;\; K << N \tag{2.1}$$

We choose to have a linear basis model, meaning that the model is defined through a linear combination of basis instances. The main reason for this choice is that linear bases can be easily learned. Also variations in the data can be often well covered by such bases. Furthermore, it is essential in our model that the number

of bases K is much less than the number of training shapes N. This condition assures that we actually model the class content which is shared across the images. If one would choose $K = N$, the training shapes could be perfectly reconstructed, but there would be no generalisation in terms of class modelling at all (Overfitting). The question which remains is how one can learn linear bases from a training set. A standard way of doing this is given by PCA. However, it is necessary to use aligned data with correspondences in order to get reasonable results. Therefore, preprocessing of our image collection will be needed.

Now having defined the deformation model, the key concept to segment the given images with the help of this model can be formulated. The idea is to use an iterative algorithm. In each step of this algorithm the segmentations of all the images shall be improved. This is achieved by restricting them to lie in the subspace of bases, which is nothing but encouraging them to have a class-like form. In the following sections the entire algorithm will be explained in detail, starting with notation and image preprocessing. In the end, variations of the algorithm are introduced.

2.3 Notation

The following symbols are continuously used in the further text and shall therefore be explained here.

N	number of examined images
K	number of bases
n	index for images ($n = 1, ...N$)
i	index for bases ($i = 1, ..., K$)
u_n	n-th 'likelihood image'
b_i	i-th basis
s_n	binary segmentation of n-th image
$\hat{s}_n$	Approximation of s_n
f, b	foreground/background label
c	colour
$p(c\|f)$	likelihood of colour given foreground

2.4 Image preprocessing

Figure 2.1: Examples of normalised and aligned images including colour padding (size: 100 x100)

Before applying the algorithm it is necessary to normalise and align the images of the given collection. This is an implicit requirement in order to derive reasonable results in the PCA. To simplify the task we assume the objects in the images to be already localised by means of bounding boxes extracted from ground-truth segmentations. In a more general approach one could think of deriving them with an objectness measure like it is done in [1]. The normalisation is carried out with respect to the larger of the two bounding box dimensions. The images are aligned to the center of the corresponding bounding boxes. To guarantee that all the images have the same size, colour padding is introduced. A neutral colour is chosen which rarely appears in the images. Examples of preprocessed images are shown in figure 2.1.

2.5 Basic Algorithm

2.5.1 Overview

The main algorithm is basically an iteration process, starting with a reasonable initialisation. Figure 2.2 shows a conceptual scheme of the algorithm, t defines the iteration variable. Notice that the scheme formulates exactly what was discussed in section 2.2. Explanations of the individual steps within the algorithm follow.

ALGORITHM

- $t = 0$: Initialisation of foreground and background regions with $\hat{s}_n(0)$, $t = t + 1$.
- $t > 0$: Iteration until only small changes from $u_n(t-1)$ to $u_n(t)$
  1. Update the appearance model with respect to $\hat{s}_n(t-1)$.
  2. Compute the 'likelihood images' $u_n(t)$ with respect to the updated colour distribution.
  3. Compute the binary segmentations $s_n(t)$
  4. Learn the deformation model and compute $\hat{s}_n(t)$
  5. $t = t + 1$ and go to step 1.

Figure 2.2: Basic algorithm

2.5.2 Initialisation

The need for an initialisation of foreground and background is given, in order to set up the starting colour distribution in the iteration process. Objects are very likely to have some parts in the center of the bounding box. Therefore, it makes sense to define a centered circular foreground region as a first guess in each of the images. The radius of this circle can be estimated experimentally.

2.5.3 Updating the appearance model

The appearance model is described through foreground and background colour distributions. The idea is to gather information about colours that are likely to be part of the objects and colours that are not.

We assume to have three colour channels in the images: red, green, blue (RGB). The distributions are build out of 3 dimensional histograms, where each dimension describes the range of values of one colour.

Basically one can imagine that the RGB-space is binned into small cubes, each of which containing the occurrence of colour combinations in that cube to be foreground or background, respectively. Therefore, the knowledge from $\hat{s}_n$ is used, which provides the current guesses of foreground and background. Notice that $\hat{s}_n(t)$ is in general a grey-level image, hence one pixel contributes always to both distributions. An exception is $\hat{s}_n(0)$ which is in binary form. In order to derive valid probability distributions $p(c|f)$ and $p(c|b)$, the histograms are normalised with the sum of occurences.

Note that there are two conceptually different ways to compute the foreground and background distributions. On the one hand we can think of creating these colour distributions over all the images. This implicitly covers a kind of class modelling, because information is shared across the images. On the other hand these distributions can be computed for each image itself, without sharing at all. In any case the regions of colour padding are not taken into account.

2.5.4 Likelihood images

The goal of this step is to compute the likelihood of each pixel (in all the images) to be foreground. The entire information is then captured in so called 'likelihood images', where the value of each pixel encodes its foreground probability. The probability is derived from the appearance model, which is described by the foreground and background colour distributions.

$$\tilde{p}(c|f) = \frac{p(c|f)}{p(c|f) + p(c|b)} \tag{2.2}$$

For each pixel, $\tilde{p}(c|f)$ denotes the foreground likelihood. $p(c|f)$ and $p(c|b)$ are the likelihood values of the respective pixel colour extracted from the foreground and background distributions. Notice that $\tilde{p}(c|f)$ results basically from a normalisation of likelihoods and is therefore also a likelihood.

2.5.5 Compute segmentations

This step builds the transition from grey-level 'likelihood images' to binary segmentations. Therefore, s_n can be derived from u_n in two ways, through Maximum Likelihood (ML) or Maximum a Posteriori (MAP) estimation. They are explained in section 2.7.

2.5.6 Learning deformation

This step of the algorithm is essential, because here the global deformable shape model is introduced. Ideally one would directly construct the segmentations s_n so that they lie in the basis subspace of the class. Because this is difficult we choose to solve the problem sequentially in two steps. First the segmentations are generated like in subsection 2.5.5, in a second step they are restricted to lie in the basis subspace. Therefore, the segmentation images are reconstructed with basis images derived from a PCA of the s_n. The resulting approximations $\hat{s}_n$ are then enforced to lie in the desired class subspace.

$$\hat{s}_n = B \cdot \alpha_n, \quad \alpha_n = \arg\min_{\alpha} ||s_n - B \cdot \alpha||^2 \tag{2.3}$$

Here B describes the set of basis images, α_n is a least squares solution. Note that given a set of segmentations and a fixed number K of bases, B and α_n are uniquely determined through the PCA.
For a technical description of the PCA including the Singular Value Decomposition, the reader is referred to the Appendix A.

2.6 Incremental learning

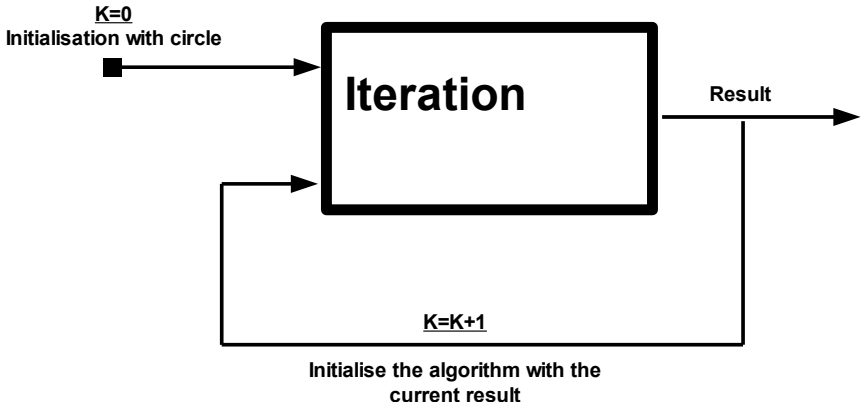

Figure 2.3: Stepwise increasing of number of bases

As described in the previous section, for the approximation of $\hat{s}_n$, we use K basis images. Note that K determines the complexity of our deformation model. A lot of bases allow the segmentations to lie in a larger subspace. Still we assume that the class subspace is small, so that $K << N$, in order to achieve proper generalisation. However, the choice of K is not straight forward, and that is why we introduce an incremental learning setting. The goal is to learn the bases one after another, while increasing the complexity of the deformation model up to a maximum number of bases.

In terms of the precise implementation, we start with $K = 0$ (model defined through mean image), and apply the algorithm, where we intialise with a circular area (subsection 2.5.2). In the next step, we increase the number of bases to $K = 1$. But this time the algorithm is initialised with the result of $K = 0$. The procedure will continue until the fixed maximum number of bases is reached. Figure 2.3 illustrates the concept.

2.7 Variations of the algorithm

So far we dealt with a very generic description of the basic algorithm like in figure 2.2. In the following, let us have a look at variations and their motivation.

2.7.1 Likelihood images and segmentations (V1)

Raw

In this algorithm setting step 3 of the basic framework (fig 2.2) is removed, no binary segmentations are considered. Therefore, in step 4 the deformation model is learned directly from the 'likelihood images' u_n. We observed that in order to achieve reasonable results for this setting, it is necessary to scale down the 'likelihood image' sizes and compute the bases from those scaled images. The reconstructed images can then be rescaled for the appearance model update.

Maximum Likelihood (ML)

Here the binary segmentation images s_n are derived through a ML estimation. Hence, the label of a pixel is assigned foreground if the foreground likelihood is higher than the background likelihood. The background labels are assigned respectively.

$$x = \arg\max_x p(c|x), \quad x = \{f, b\} \tag{2.4}$$

Maximum a posteriori (MAP)

So far only unary terms were used to derive s_n, meaning that every pixel in each image was considered for itself. Graph-cut (for details refer to [3]) additionally allows to take pairwise terms into account while deriving a binary segmentation. So neighbourhood knowledge between the pixels of an image can be used as well to compute the segmentations. In addition, we are able to introduce prior knowledge and data dependent pairwise terms. As Graph-cut finds the optimal labellings of pixels through an energy minimisation (section 2.1), we need to define the energy terms in this application.

unary energy terms The unary potentials $\Upsilon(j_n)$ in each of the images are defined as the foreground likelihoods u_n or the ML estimation of u_n. The latter is used for the evaluations in chapter 3.

pairwise energy terms Here we consider two types of pairwise relations between pixels. On the one hand we establish a smoothness prior $\Lambda(j_n, l_n)$. It makes sure that neighbouring pixels have same labels. On the other hand we introduce data dependent pairwise terms $\Pi(j_n, l_n)$, where we can incorporate local features like edges. So we encourage that the foreground pixels follow the object contours. For more details about this second pairwise term, the reader is referred to the Appendix B.

Summarised, the final energy which has to be minimised by Graph-cut for a single image can be formulated as follows

$$E_n = \upsilon \sum_{j_n} \Upsilon(j_n) + \lambda \sum_{j_n \neq l_n} \Lambda(j_n, l_n) + \pi \sum_{j_n \neq l_n} \Pi(j_n, l_n), \quad n = 1, ..., N \tag{2.5}$$

The indices j_n and l_n number all the pixels in the n-th image. Therefore, e.g. $\Lambda(j_n, l_n)$ indicates the pairwise smoothness potential between pixels j_n and l_n. Note that the individual potentials can be weighted differently.
As Graph-cut derives the optimal binary segmentations while incorporating likelihood as well as prior knowledge (smoothness), the described procedure corresponds to a MAP estimation.

2.7.2 Colour distribution (V2)

An alternative variation of the algorithm affects the appearance model. Primarily one can think of having a global foreground and background colour distribution over all images. This would implicitly lead to class modelling and might lead to higher generalisation. In contrast to this, one can also imagine to model appearance for single images. Therefore, we consider foreground and background distributions individually for each image.

2.7.3 Incremental Learning (V3)

This dimension of variation describes whether the algorithm is executed with or without incremental learning. Comparing the two performances we can see if incremental learning can support the class modelling process.

2.7.4 Deformation Model (V4)

Finally one can think of varying step 4 of the basic algorithm (figure 2.2). Therefore, one could eliminate this step completely, meaning that we would not incorporate any global deformable shape model. This can be interesting in order to analyse the effects of the deformation model on the segmentations.

Chapter 3

Experiments and Results

3.1 Dataset

The performance of the proposed algorithm is evaluated on the Weizmann horse dataset. It consists of 328 images, each one representing a single horse, whose head is pointed to the left. The horses show a great variety in colour, pattern and also in pose. Nevertheless, we notice that a lot of images show dark horses whereas white horses appear rarely. Also there are a lot of horses standing in static position with front and back feet parallel to each other. Still there are a few which take dynamic poses such as running or poses where the head is bowed to the ground. Examples of the original dataset images are shown in figure 3.1, whereas the aligned images were already demonstrated in figure 2.1.

Figure 3.1: Weizmann horses

3.2 Software environment

The complete implementation of the algorithm with all its variations is realised in MATLAB R2010a. Besides an external Graph-cut code ([3],[4],[2]) provided by Olga Veksler and some additional functions from Mukta Prasad, all the programming was done by the author of this work.

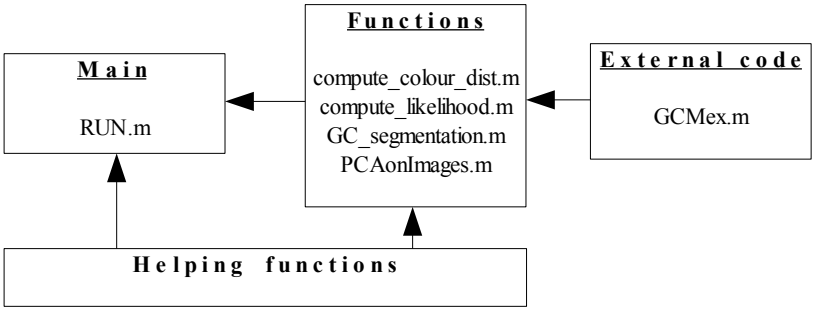

Figure 3.2: Scheme of the algorithm implementation with MATLAB (the abbreviation GC describes Graph-cut)

Program structure of the algorithm The code is structured modular, which means that new functions can be easily added, and existing functions can be edited with little effort. Therefore, all the important functions are divided into separate m-files. An example: If one wants to use another colour distribution than a RGB histogram, e.g. a Gaussian Mixture Model, the changes would only affect one single m-file (compute_colour_dist.m).

3.3 Analyis of ground-truth segmentations

Before we evaluate the algorithm, it is useful to analyse the results of a PCA on ground-truth segmentations in terms of energy and modes of variation. Primarily this allows us to have some insight into the actual deformation model which is used for segmentation. Analogous the steps explained in chapter 2, a preprocessing is done to normalise and align the ground-truth segmentations. The difference is that we align with respect to the centroid, which is defined through the center of mass of the foreground pixels (not center of bounding box). The Weizmann dataset is used for this analysis.

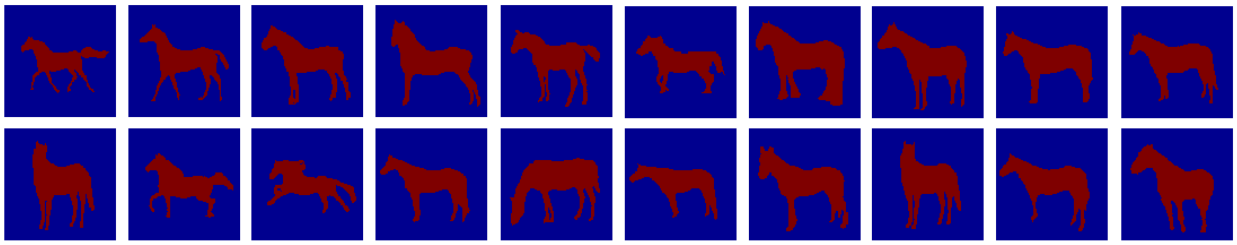

Figure 3.3: Examples of ground-truth images after normalisation and alignment.

3.3.1 Energy

We analyse the energy of the bases. Energy is used as a synonym to variance here. Each of the bases covers a certain amount of variance in the data. Thereby they are ordered such that the first covers the most, and the

last covers the least. Notice that this does not tell quantitatively how much energy is within each basis. In the worst case each of the them would cover the same amount of variance. For our segmentation images this would mean that there was no significant information shared across them. To draw reasonable conclusions

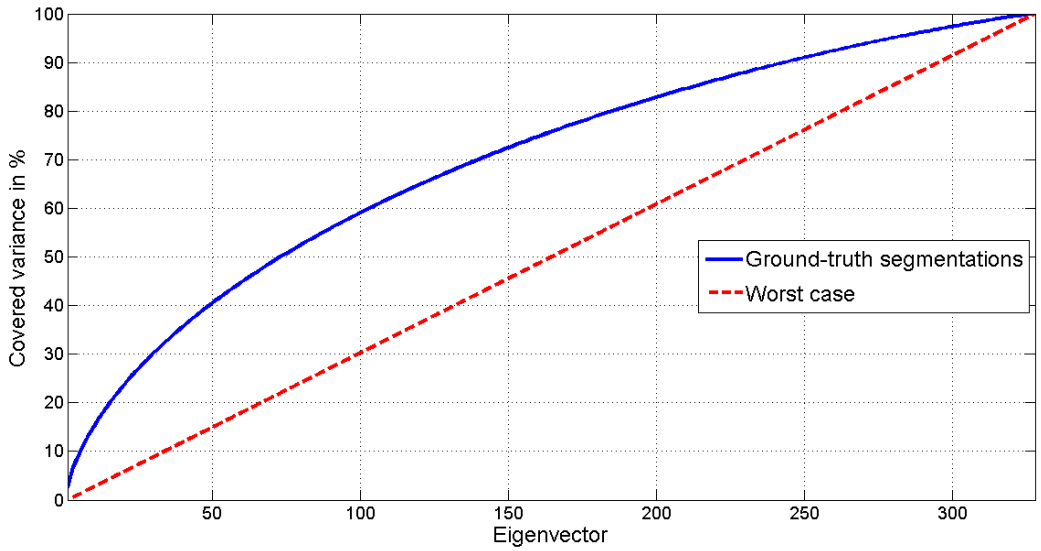

Figure 3.4: Cumulative sum of eigenvalues resulting from PCA of ground-truth segmentations, and hypothetical worst case scenario. PCA on 328 images of size 100 x 100.

one can consider the eigenvalues of the PCA. This is interesting because they quantify the variance of the corresponding eigenvectors. Ideally we plot the cumulative sum over the normalised eigenvalues[1]. Figure 3.4 shows such a plot for the given PCA of ground-truth segmentations. Compared to the worst case, which represents equal variance in each of the eigenvectors, the eigenvalue distribution of the ground-truth segmentations shows a meaningful behaviour. Considering the first 100 eigenvectors which are about 30% of all the eigenvectors, already 60% of the variance in the data is covered. This result indicates that a lot of information is in the first few bases.

3.3.2 Modes of variation

Besides the eigenvalues also the eigenvectors (bases) should be examined. That way it is easier to develop an intuition about what we can expect in our algorithm. To visualise the information content of the i-th basis image b_i, we consider

$$m_{i+} = \overline{I} + \delta \cdot b_i \quad and \quad m_{i-} = \overline{I} - \delta \cdot b_i, \quad i = 1, ...K \ ,$$

where $m_{i\pm}$ represents the positive or negative variation in the i-th basis image, $\overline{I}$ describes the mean image, δ defines the strength of variation, and K defines the number of basis images. Figure 3.5 shows the first three modes of variation resulting from the PCA on the ground-truth segmentations. In the upper images the positive variations m_{i+} are illustrated, the lower ones represent m_{i-}. Interestingly, reasonable interpretations of the different basis images exist. The first basis could be interpreted as a dynamic motion: the

[1] $\hat{\lambda}_i = \frac{\lambda_i}{\sum \lambda_i}$, λ_i: eigenvalue, $\hat{\lambda}_i$: normalised eigenvalue

variation goes from a still standing horse to a running one. The second basis could explain position of the head and whether front and back feet are standing parallel or more in a walking position. Lastly, the third basis indicates the notion of the head pointing up or down.

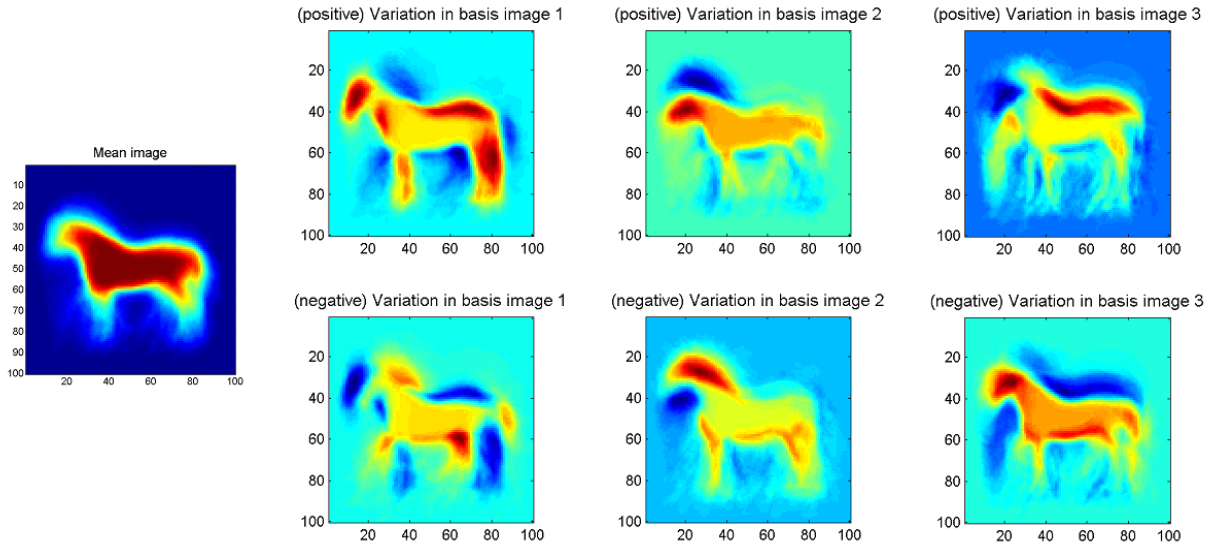

Figure 3.5: Modes of variation: mean image and $m_{i\pm}$ for $i = 1, 2, 3$, δ = 30

Summarised, from the energy analysis it follows that the first few bases contain a significant amount of information. In addition, we can assign meanings to the basis images. The deformation model should therefore be able to model the class content.

3.4 Parameter settings

For the further evaluations the following major parameters of the algorithm are set empirically. For this purpose 'GC_segmentation.m' was tested on various images, decoupled from the algorithm. The radius r is described as a fraction of half of the images length.

Intialisation radius	$r = 0.3$
Number of bins in RGB-histogram	$nBins = 20$
Graph-cut: unary weight	$\upsilon = 0.8$
Graph-cut: smoothnes weight	$\lambda = 1$
Graph-cut: data dependent pairwise weight	$\pi = 1$

3.5 Qualitative evaluations

Various experiments are carried out now, and the algorithm is evaluated in terms of images like final segmentations or reconstructions. It is shown how the different settings influence the quality of the results. For explanations of V1,V2,... the reader is referred to section 2.7.

3.5.1 Raw vs. MAP

V1	V2	V3	V4
-	Individual colour distribution	with incremental learning	with deformation model

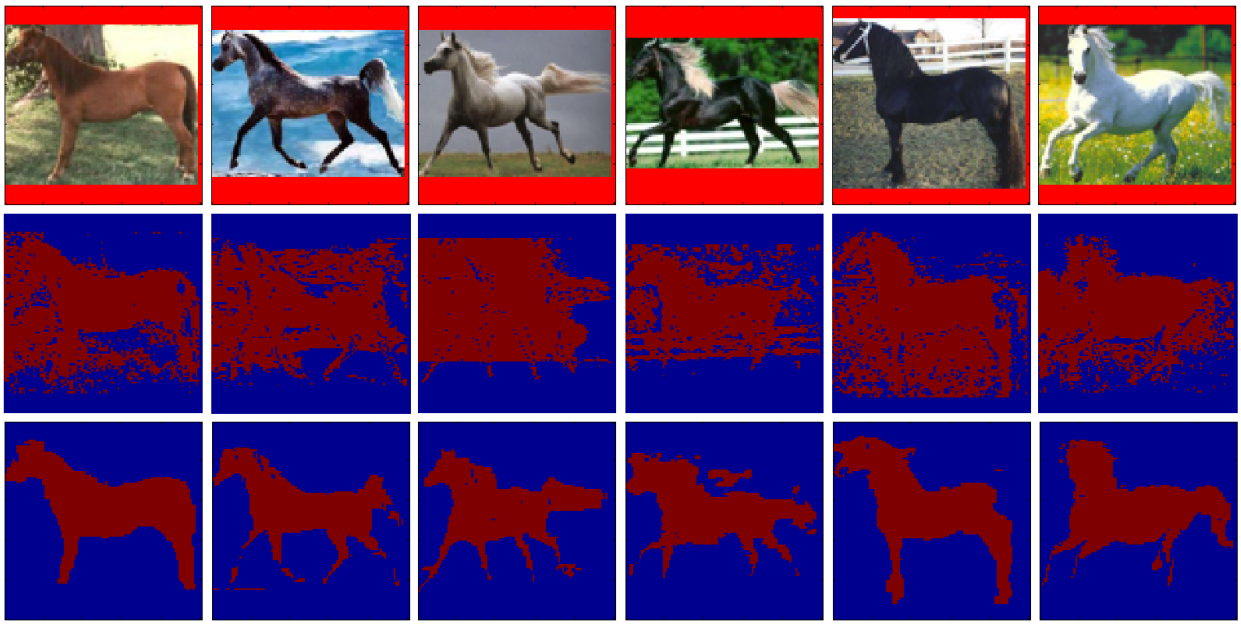

Figure 3.6: Raw with scaling of 0.08 (second row) vs. MAP (third row)

In this experiment the effects of different inputs for the deformation model are illustrated. In the raw setting we are building the bases directly from the 'likelihood images'. We note that only unary terms are used, meaning that neighbourhood relations are ignored. No prior knowledge is used. Therefore, we can observe in figure 3.6 that the Raw segmentations look very noisy and are not at all capable of deriving clean object boundaries. Definitely, too many regions are declared to be foreground. Looking at the MAP estimation we find huge improvements. A lot of noise is removed, so that clean shapes result. In addition, we note that the shapes follow most of the object contours.

3.5.2 Global vs. Individual colour distribution

V1 MAP	V2 -	V3 with incremental learning	V4 with deformation model

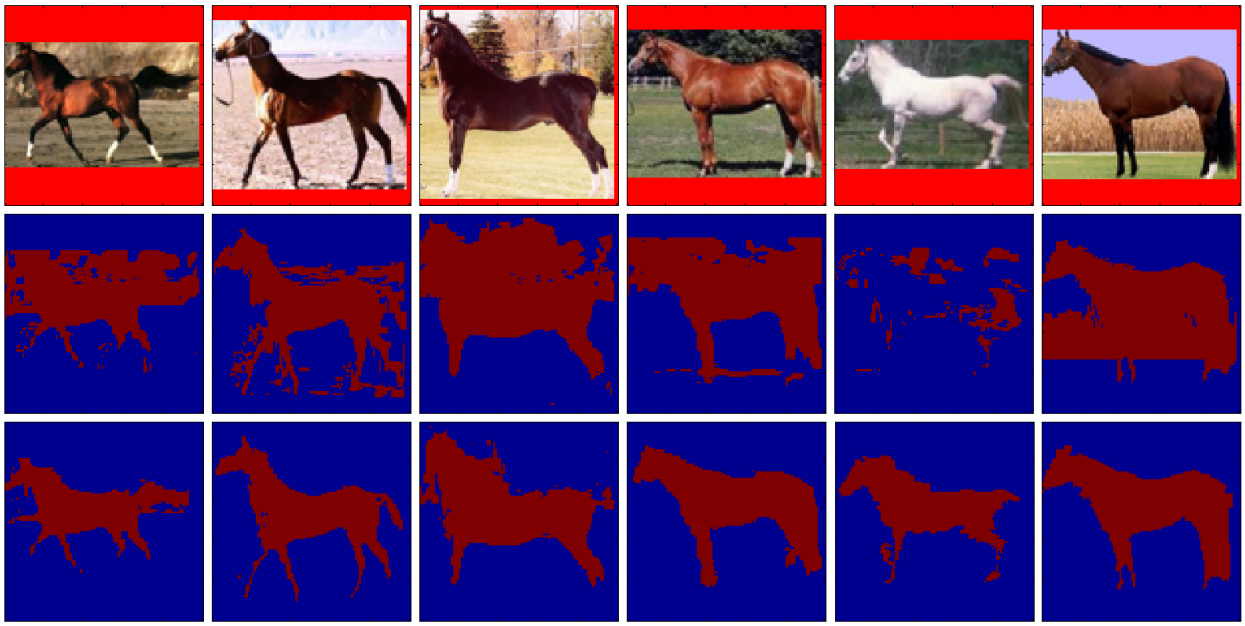

Figure 3.7: Global (second row) vs. Individual (third row) colour distribution

As the whole algorithm relies on the appearance model, it is interesting to take a look on variations in this dimension. The main message here is that we observe better results with individual than with global colour distributions. Especially white horses which appear as some sort of outliers, are badly explained in terms of global colour distributions. But also in images with darker horses global appearance models lead to minor results, as they tend to classify background regions as objects.

3.5.3 Effects of incremental learning

V1 MAP	V2 Individual colour distribution	V3 with incremental learning	V4 with deformation model

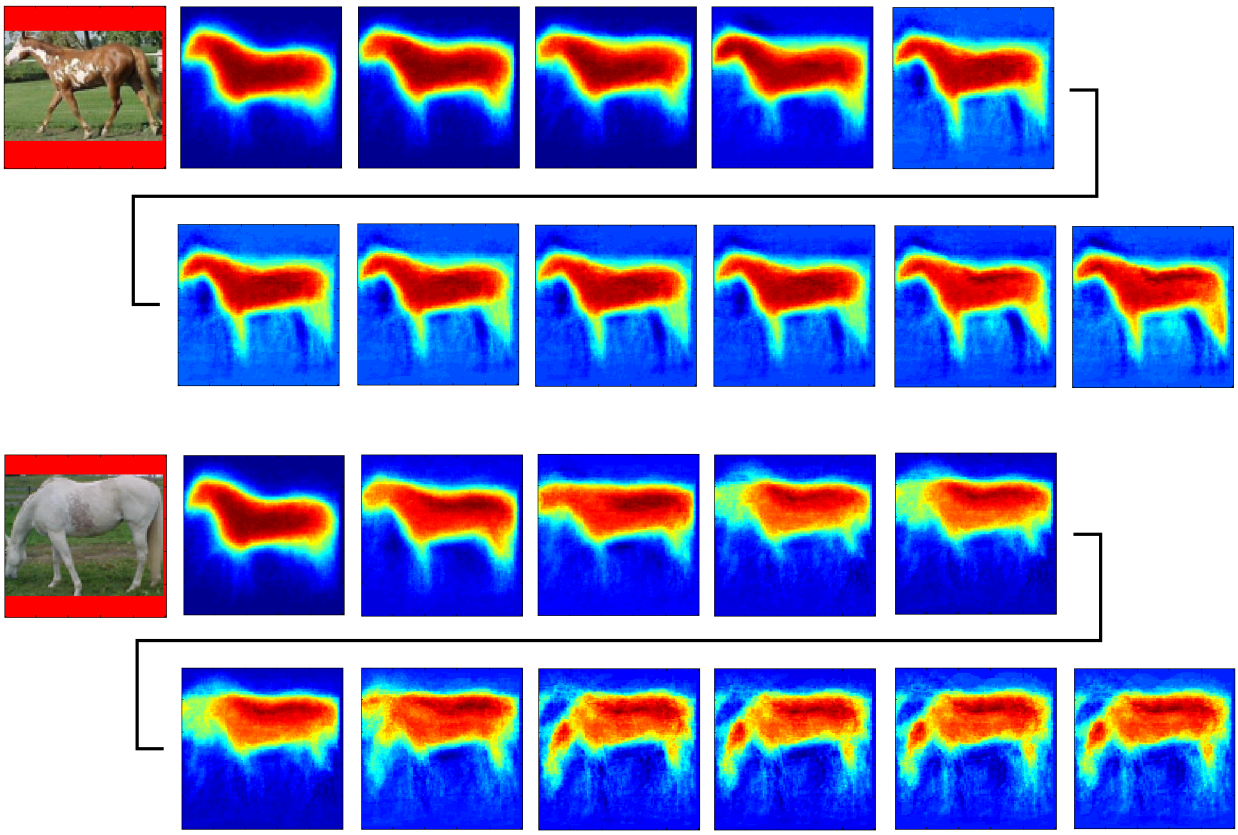

Figure 3.8: Effects of incremental learning (K=0,1,...,10) shown in $\hat{s}_n$ for each K

The idea of incremental learning is illustrated within figure 3.8. For two horses the development of the reconstructed segmentation $\hat{s}_n$ is shown during the procedure described in section 2.6. Starting from the top left reconstructed segmentation where K=0 (mean image) we see the evolution of the deformation model until K=10 on the bottom right. Therefore, these images show what happens when the complexity of the model is stepwise increased. Still, we have to be aware that it *makes no sense to have an arbitrary complex model* because generalisation would be lost. Nevertheless, for *small numbers* of bases it can be useful to increase the complexity. We can actually observe the effects of a more sophisticated model in terms of structural changes in the horse segmentations. In the upper sequence we see how a notion of feet is established. The lower sequence shows how we learn that the horse has its head turned down.

3.5.4 What is the contribution of the Deformation model?

V1	V2	V3	V4
MAP	Individual colour distribution	with incremental learning	-

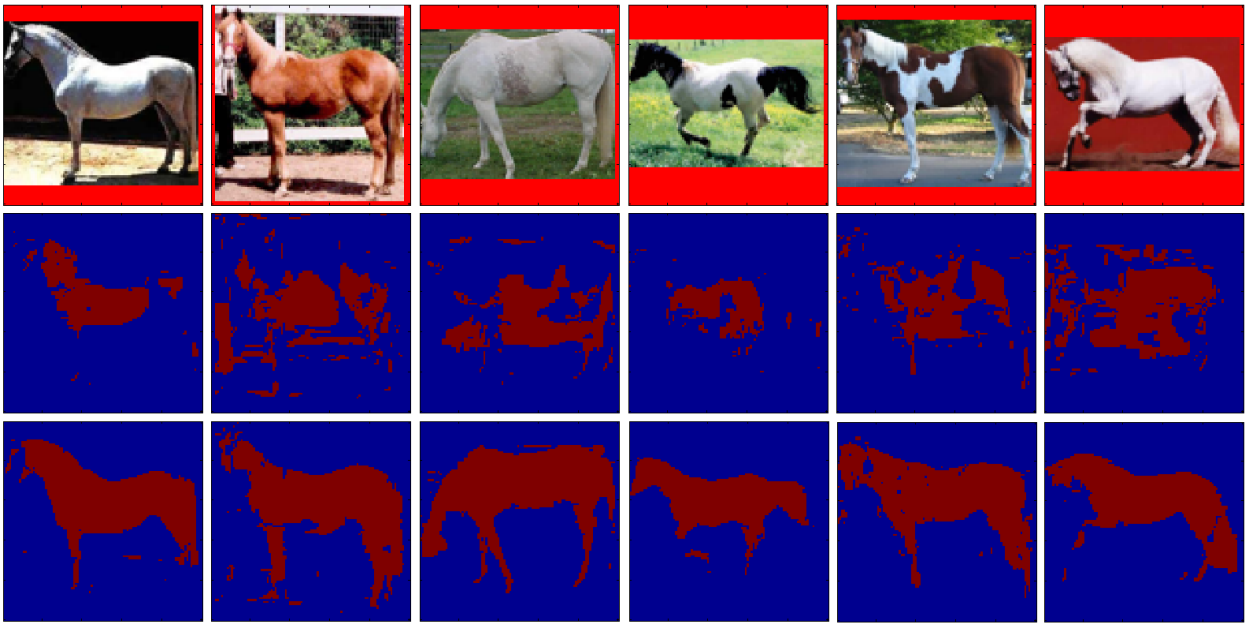

Figure 3.9: without (second row) vs. with (third row) use of deformation model

Now that we saw MAP, individual colour distributions and incremental learning lead to good results, we want finally examine the use of the deformation model. Therefore, as described in subsection 2.7.4, we run the algorithm without restricting the segmentations to lie in the basis subspace but with our best configurations (MAP, individual colour distribution). We compare the results with our best settings including deformation modelling.

The images in figure 3.9 justify the use of a deformable shape model. The final segmentations incorporating deformation are way more meaningful than the others. Especially the horse in the fourth column shows major improvements. Although it has two very different colours and is in a running pose (not typical with respect to the dataset), the deformation model allows to segment it properly.

3.6 Quantitative evaluations

As a quantitative measure for the performance of the algorithm, the accuracy of the final segmentations with respect to the ground-truth is computed. Thereto, we calculate the percentage of all correctly classified (foreground/background) pixels with respect to the total number of pixels in all images.

$$Accuracy = \frac{correctly\ \ classified\ \ pixels}{total\ \ number\ \ of\ \ pixels} \cdot 100\% \tag{3.1}$$

Obviously, the colour padding regions are not considered in the computation. The results are reported in tables 3.1-3.3. All possible variations which were explained and shown can be compared. Note that for the Raw setting the segmentations are derived with a ML estimation in the very last iteration step. We see that the tendencies from the qualitative results are proved. MAP leads as well as individual colour distributions to better results. Moreover, we can retain that incremental learning improves the segmentations with individual colour distributions (except raw). Most importantly, with the exception of the raw algorithm setting, we can also claim that our deformation model enhances the segmentations significantly.

Colour distribution	without deformation model	without incremental learning					with incremental learning (up to $K = 10$)
		$K = 5$	$K = 10$	$K = 20$	$K = 50$	$K = 100$	
Individual	**72.68**	71.13	72.03	72.41	72.47	-	72.25
Global	73.42	73.37	73.55	73.39	73.37	-	**73.80**

Table 3.1: Accuracy for Raw with scaling of 0.08

Colour distribution	without deformation model	without incremental learning					with incremental learning (up to $K = 10$)
		$K = 5$	$K = 10$	$K = 20$	$K = 50$	$K = 100$	
Individual	74.62	83.36	83.69	83.26	81.12	-	**86.24**
Global	73.35	73.55	74.29	74.85	**75.07**	74.65	72.80

Table 3.2: Accuracy for ML estimation

Colour distribution	without deformation model	without incremental learning					with incremental learning (up to $K = 10$)
		$K = 5$	$K = 10$	$K = 20$	$K = 50$	$K = 100$	
Individual	77.12	86.62	85.92	85.17	84.00	-	**87.40**
Global	74.78	74.82	74.74	75.03	**75.38**	75.11	74.26

Table 3.3: Accuracy for MAP estimation

As a reference, we can compare the obtained segmentation accuracies with those of ClassCut (table 3.4). Although the accuracy of our approach looks a bit better, it is important to see under which conditions this comparison is made. For one thing, ClassCut deals also with the problem of object detection and

ClassCut	Our approach
86.2%	87.4%

Table 3.4: Best segmentation accuracies from ClassCut and our approach

therefore the extraction of bounding boxes. We assume these to be given from ground-truth in our problem. For another, ClassCut uses very sophisticated and advanced energy terms, whereas we basically refer to Principal Component Analysis and Graph-cut. Therefore, the obtained results are quite amazing.

Chapter 4

Discussion and Conclusion

4.1 Interpretation of the results

For the proposed algorithm it was demonstrated that individual colour distributions lead to better final segmentations than the global colour distributions. At first sight, this result may seem contradictory, because one could expect higher generalisation with a global appearance model. The reason why the performance is still lower, can be explained by taking a closer look at the dataset. In our Weizmann dataset the foreground objects are horses. Rather than colour, the shape is the main attribute that defines their appearance as horses. While they differ very much in colour (brown, black, white, mixed) from one image to another, the colours don't change that much within a single horse. Additional problems occur because often there are white background regions in the images. This prevents white horses from being detected at all if we use a global colour distribution. However, global appearance models should not be underestimated in general. Thinking of an arbitrary class object of another dataset, appearing in very similar colours, a global distribution will improve the segmentation.

Another interesting observation from chapter 3 is that the MAP estimation delivers the best results. However, the results in the best configuration (with individual colour distribution and incremental learning) do only differ in 1% from the ML estimation. This indicates that the true magic of the algorithm lies in the unary likelihood terms, whereas the smoothnes prior and the data dependent pairwise terms fulfill the purpose of small additional optimisation. Moreover, we notice that the Raw algorithm setting does not lead to significant improvements at all. The main reason is that our deformation model has problems to handle grey-level images as inputs. More precisely the Principal Component Analysis of a set of grey-level images does not result in reasonable bases. Ideally binary segmentations should be used, like for the ML and MAP estimation.

Finally we want to discuss our deformation model. Several observations in context with the final segmentations, and the analysis of energy and modes of variation have shown that the linear basis model is suitable for the segmentation task and leads to major results in comparison to an approach without any global shape model. The bases can definitely describe the class information which is shared across the images, although we use a sequential implementation where we first generate the segmentations and restrict them afterwards.

4.2 Possible Improvements

- The need for an initialisation of the algorithm can be seen as a negative point. If we consider another dataset, the radius of the circular initialisation region has to be tuned first. It might be interesting to experiment with other initialisations. One could think of collecting prior knowledge about the object location beforehand to improve this process.
- The main focus of this work was to prove that a global deformation model is able to support the segmentation process. Therefore, computation time was not an issue. As the algorithm works quite slowly, especially when we embed incremental learning, surely there is room for improvement concerning this aspect.
- An issue is the embedding of pairwise terms in the algorithm. To be able to use neighbourhood information, Graph-cut with its pairwise energy terms was introduced. Clearly with PCA and the computation of 'likelihood images' the unary terms, which consider only the pixels themselves, play the dominant role. One has to realise that the 'likelihood images', via the global deformation model, are defined over all images of the dataset, whereas the smoothness prior and the edge dependent pairwise terms are only defined for each image itself. Therefore, updating the pairwise terms over all images (like in [1]) could support the segmentation process.
- Alternative appearance models could probably improve the algorithm. The RGB histogram used in this work is a very simple solution, alternatively colour distributions can be built with Gaussian Mixture Models. In general one could also think of using a completely different appearance model, like e.g. a characterisation with descriptors.

4.3 Conclusion

To solve the task of simultaneous object instance segmentation, a simple iterative framework was introduced. The key idea was to use a global deformable shape model in order to improve the segmentation. Various algorithm settings were discussed, evaluated and compared. Finally the results showed that a linear basis deformation model can improve the segmentation and can therefore be regarded as a good shape model for the task. The best results for the Weizmann horse dataset were achieved with a Maximum a posteriori estimation, individual colour distributions and incremental learning.

Appendix A

Principal Component Analysis of images

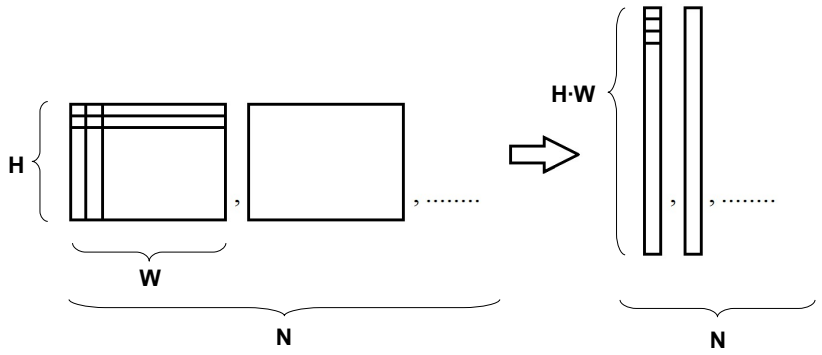

Figure A.1: Transformation of images into vectors

1. Transform the N images into N column vectors. Like explained in figure A.1, H defines the height and W the width of an image. Form a data matrix X from the column vectors.
2. Normalise the data by subtracting the mean.

$$\tilde{X} = X - \overline{X} \tag{A.1}$$

3. If we assume $N << H \cdot W$, then computing the covariance matrix as in equation A.2 reduces computation time (procedure also known as PCA trick)

$$C_x = \frac{1}{H \cdot W - 1} \cdot \tilde{X}^T \tilde{X} \tag{A.2}$$

4. Compute the Singular Value Decomposition

$$C_x = U_{red} S V_{red}^T \tag{A.3}$$

5. The final decomposition of the normalised data

$$\tilde{X} = \sqrt{diag(S)} \cdot U \tag{A.4}$$

$$U = \tilde{X} U_{red} \tag{A.5}$$

where $\sqrt{diag(S)}$ describes a vector of eigenvalues, resulting from the square roots of diagonal elements of S. The column vectors of U are the eigenvectors. The **principal components** or **basis vectors** are defined as the eigenvectors with the largest K corresponding eigenvalues.

Appendix B

Data dependent pairwise terms

We use data dependent pairwise terms in order to encourage that pixels have different labels along edges. Keep in mind that a negative potential between two pixels leads to lower total energy if it is cut. Therefore, we can formulate the following procedure to establish the edge dependent pairwise terms for a single image.

1. Compute the edge image from the original RGB image (e.g with canny edge detector).
2. Compute the distance transform of the edge image, which indicates at each position the distance to the nearest edge pixel.
3. For all directly neighbouring pixel pairs, if the two pixels have different distance transform values, we define a negative potential between them. The weight of the absolute potential decays exponentially with the distance from the edge.

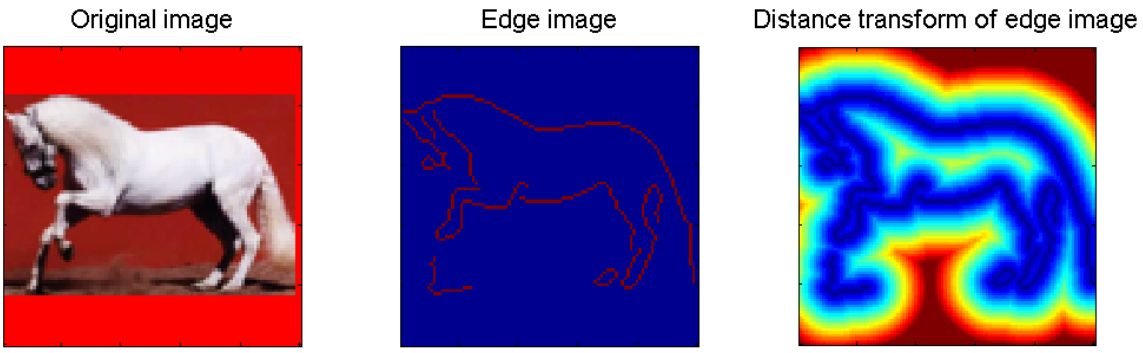

Figure B.1: Examples of edge image with corresponding distance transform

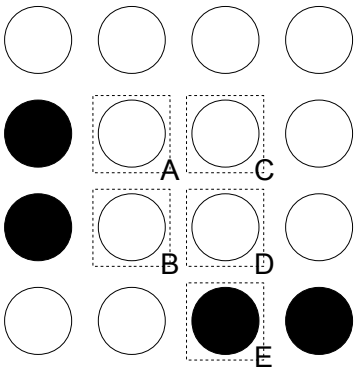

Figure B.2: Example of pixel structure in an edge image, circles indicate pixels, black circles indicate edges

Looking at figure B.2, we can discuss some example potentials.

- $\Pi(A, B) = 0$, because A and B have equal distance to the nearest edge pixel. In other words, the pixel pair is parallel to the edge.
- $\Pi(C, D) > \Pi(D, E)$, because the pair (D,E) is closer to the edge and therefore gets a more negative potential than (C,D). We notice that when exactly one of the pixels lies on an edge the absolute edge dependent pairwise potential is highest.

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Classcut for unsupervised class segmentation. *Computer Vision - ECCV*, 2010.

[2] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2004.

[3] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE transactions on PAMI*, 2001.

[4] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2002.