DOI:10.15774/PPKE.ITK.2013.005

Fast Content-adaptive Image Segmentation

PhD dissertation

Balázs Varga

Scientific advisor
Kristóf Karacs, PhD

Consultant
Tamás Roska, DSc
ordinary member of the
Hungarian Academy of Sciences

Doctoral School of Multidisciplinary
Engineering Sciences
Faculty of Information Technology
Pázmány Péter Catholic University

Budapest, 2012

DOI:10.15774/PPKE.ITK.2013.005

DOI:10.15774/PPKE.ITK.2013.005

Acknowledgements

I would like to thank my scientific advisor *Kristóf Karacs* for guiding and supporting me over the years. You have set an example of excellence as a mentor and instructor.

I am grateful to *Prof. Tamás Roska, Prof. Árpád Csurgay, Prof. Péter Szolgay*, and *Judit Nyékyné Gaizler, PhD* for giving me encouragement, words of wisdom, and the opportunity to carry out my research at the University.

The support of the Swiss Contribution and Tateyama Laboratory Hungary Ltd is kindly acknowledged.

During the first years I got plenty of ideas and concepts in the field of computer vision and machine learning from a fellow PhD student *Vilmos Szabó*, furthermore, many thanks go out to my other colleagues in the Doctoral School for their friendship and/or for the knowledge and the amazing times we spent together: *Csaba Nemes, László Füredi, Ádám Balogh, Kálmán Tornai, Dávid Tisza, Mihály Radványi, Andrea Kovács, Róbert Tibold, Norbert Bérci, Gergely Feldhoffer, Gábor Tornai, Tamás Zsedrovits, Ádám Rák, József Veres, Attila Stubendek, Zóra Solymár, Péter Vizi, Miklós Koller, András Kiss, András Horváth, Gergely Treplán, Dániel Kovács, Ákos Tar, László Laki, Tamás Fülöp, Zoltán Tuza, György Orosz, János Rudán, Anna Horváth, Zoltán Kárász, Ádám Fekete, András Gelencsér, Tamás Pilissy, Balázs Karlócai, Dániel Szolgay, András Bojárszky, Éva Bankó, Gergely Soós, Ákos Kusnyerik, László Kozák, Domonkos Gergelyi, István Reguly, Dóra Bihary, Petra Hermann, Balázs Knakker, Antal Tátrai, Emília Tóth*, and *Csaba Józsa*.

DOI:10.15774/PPKE.ITK.2013.005

I wish to thank the postdoctoral fellows for keeping that special spark of the Doctoral School alive: *András Oláh, Zoltán Nagy, Miklós Gyöngy, Kristóf Iván*, and *György Cserey*.

Special thanks go out to *Viktória Sifter, Lívia Adorján, Mária Babiczné Rácz, Mária Körmendyné Érdi* and *Péter Tholt* who were always kind to offer instant solutions instead of additional complications in the maze of everyday bureaucracy.

Last and most importantly, I can not really express the level of gratitude to my *Family*, my girlfriend *Dóri*, my *Friends*, and my *RV-NB10*, who were always there for me to share the good times and the hard ones as well.

DOI:10.15774/PPKE.ITK.2013.005

Abstract

My work aims at making a step towards efficient image understanding through the possibilities of fusing the top-down and the bottom-up approaches. In this dissertation I put the focus on the investigation of bottom-up image segmentation and the points where injection of top-down knowledge into a data-driven segmenter is possible.

The first problem I have been working on was the fast segmentation of high resolution images. The motivation for using high resolution images is that this way one can obtain more information from the segmentation output, with objects or object details that otherwise could not be retrieved due to their small extent. Identification of a higher number of details can enhance the robustness of recognition and classification and it can also provide additional cues that top-down knowledge could be applied to. The downside of increasing the physical resolution, i.e. the number of pixels is that the amount of data to be processed grows, which has a negative impact on the running time. To overcome this problem, I constructed and implemented a framework that works in a data-parallel way, and therefore it can efficiently utilize the powerful computational capabilities of many-core environments. The principal idea of the system was inspired by the mean shift algorithm, which I extended with a quasi-random sampling scheme for further acceleration, and a cluster merging procedure to reduce over-segmentation. In addition, I proposed a method (named abridging) to reduce the overhead caused by parallelization.

The second problem I addressed was making an adaptive image sampling scheme, in order to take the local content of the image to be segmented into account. I had two simultaneous goals. On one hand, to eliminate superfluous computations for homogeneous regions with minimal content,

DOI:10.15774/PPKE.ITK.2013.005

thus to keep the amount of time required for the segmentation to a content-dependent minimum. On the other hand, to preserve an output quality similar to that of the naïve version, which is applied to every single input pixel without using any kind of sampling. To achieve this, I need not only to determine the number of samples required to maintain a certain segmentation quality, I also have to face the difficulty of finding a good spatial distribution of the samples. I developed an automated mechanism that is capable of solving this task. Additionally, the scheme uses a single-parameter for both the selection of sample candidates and for the registration of the strength of the bond between the pixels and their clusters. This way, the representation of the system remains compact, which enables the segmentation of large images as well. Furthermore, this bond confidence strategy enables each pixel to be associated with the most similar class, with respect to its spatial position and color. I designed the adaptive sampling method to fit into the realized parallel framework.

DOI:10.15774/PPKE.ITK.2013.005

Összefoglalás

Célom, hogy doktori munkámmal előremozdítsam a különböző modalitású információk fúziójára alapuló képi értés tudományterületét. Disszertációmban a hangsúlyt egy olyan bottom-up elvű, moduláris szegmentáció kutatására helyeztem, amely a klaszterezés folyamata során létrejövő osztályhierarchia különböző szintjein és pontjain ad lehetőséget top-down elvű, szemantikus, illetve feladat specifikus információk befecskendezésére.

Elsőként egy olyan eljárást dolgoztam ki, amely gyorsan képes magas felbontású képek szegmentációjára. A magas felbontás használatát az indokolja, hogy segítségével a szegmentáció kimenetéből olyan részleteket is kinyerhetünk, amelyek méretükből adódóan kisebb pixelszámú képeken nem jelennek meg. Ezen többlet információk elősegítik a különböző felismerési-, és klasszifikációs feladatok pontosabb működését, és kiszélesítik a top-down módszerekkel elérhető tudás felhasználásának lehetőségeit. A fizikai képméret növelésének hátránya, hogy a feldolgozandó adatok mennyisége is nő, amely negatív hatást gyakorol a szegmentációs algoritmusok futási idejére. Ezen probléma áthidalására olyan keretrendszert konstruáltam, amely párhuzamosított belső szerkezetéből kifolyólag ki tudja használni a sokprocesszoros számítógép architektúrákban rejlő magas számítási kapacitást. A keretrendszer magját a mean shift algoritmus inspirálta, melyet a további sebességnövekedés érdekében kiegészítettem egy kvázi-véletlen elven működő mintavételezési eljárással, illetve olyan osztályösszevonó lépést alkottam hozzá, amely hatékonyan képes a túlszegmentáció csökkentésére. Mindezeken felül bevezettem egy eljárást, amely csökkenti a párhuzamosításból fakadó többletmunkát.

DOI:10.15774/PPKE.ITK.2013.005

A második kutatási területem egy olyan, adaptív mintavételezési eljárás megalkotására fókuszált, amely a kép tartalmának (content) lokális jellemzői alapján dolgozik.

Kutatásomnak két célja volt: egyfelől a szegmentáció futási idejének minimalizálása a képtartalom függvényében, egyszersmind kimenet minőségének azonos szinten tartása a naiv (tehát az összes képponton dolgozó, mintavételezést nem tartalmazó) eljárással. Ezen célok teljesítéséhez nem csak a mintavételezéshez használt elemek számának, hanem ezek topografikus pozícióinak meghatározása is szükség van. A problémák megoldására egy autonóm módon működő eljárást alkottam, amely ezen feladat megoldásán túl egy darab paraméter értéke alapján valósítja meg a mintavételezést és ebben tárolja az osztályok és a pixelek között fennálló kötés erősségét is. Ezáltal a rendszer reprezentációja tömör módon képezhető le, így mód nyílik magas pixelszámú képek szegmentációjára is. Ez a stratégia arra is lehetőséget biztosít, hogy a pixeleket a színük és térbeli pozíciójuk alapján hozzájuk leginkább hasonlító osztályokhoz rendelhessük hozzá. Ezen adaptív módszert a fent ismertetett párhuzamos keretrendszerbe illesztettem be.

DOI:10.15774/PPKE.ITK.2013.005

Nothing in this world can take the place of persistence. Talent will not; nothing is more common than unsuccessful people with talent. Genius will not; unrewarded genius is almost a proverb. Education will not; the world is full of educated derelicts. Persistence and determination alone are omnipotent.

— *Calvin Coolidge*

DOI:10.15774/PPKE.ITK.2013.005

DOI:10.15774/PPKE.ITK.2013.005

Contents

DOI:10.15774/PPKE.ITK.2013.005

CONTENTS

DOI:10.15774/PPKE.ITK.2013.005

DOI:10.15774/PPKE.ITK.2013.005

CONTENTS

DOI:10.15774/PPKE.ITK.2013.005

List of Figures

DOI:10.15774/PPKE.ITK.2013.005

LIST OF FIGURES

DOI:10.15774/PPKE.ITK.2013.005

List of Tables

DOI:10.15774/PPKE.ITK.2013.005

LIST OF TABLES

DOI:10.15774/PPKE.ITK.2013.005

Chapter 1

Introduction

The automation of tasks that can increase the quality, or extend the duration of human life has been under permanent and heavy research for a long period of time. Such tasks include, but are not restricted to jobs that are simply too monotonous (e.g. surveillance, or 24/7 quality assurance of mass produced items moving on a conveyor belt) or can not be done by humans because of biological reasons (e.g. flight or automotive navigation tasks or medical imaging). In many of these problems we use visual data partially or exclusively, thus the accurate processing, or even more, the understanding of visual information is inevitable for the refinement of the next generation of these machines.

Of course, for the automation of complex human activities, machines and algorithms need to be equipped with a sensory-algorithmic arsenal, somewhat similarly to the senses we humans possess. Many of the more complex tasks that await to be mechanized are based not just on the processing, but on the "understanding" of visual information. The difficulty lies in the fact that besides the "sensory" input obtained by our eyes, the human brain uses complex cognitive information during the interpretation of the scene it sees.

For example, we easily identify a tennis ball in case we have already seen one before. Looking at a bag of balls that have a similar color, we can grab one without any hesitation, even though the boundaries of neighboring balls may not be clearly visible due to poor lighting conditions. This is because we *know* something about the size and shape of such an object.

Interpreting this procedure in the language of image understanding, we can make two observations. The first is that high-level metadata (having *a priori* knowledge on

DOI:10.15774/PPKE.ITK.2013.005

the physical properties of a ball) can highly aid the accuracy of execution. On the other hand, a top-down approach can not succeed without using information provided by a complementary bottom-up processing (seeing the pile of balls, utilizing the retina channels to extract low-level information), because it is the fundamental basis which high-level information is applied upon.

Being one of the most successful and straightforward sources to use, image understanding has always been inspired by the human visual system. As of today, many of the most successful algorithms in the field of segmentation (inside the broader area of computer vision) utilize certain combinations of top-down and bottom-up approaches. Just as we know plenty about how the human visual system works and processes low-level visual information, we also know how to efficiently conduct certain image processing tasks from the pixel level, in a bottom-up manner. However, just like in the case of neurobiologists and psychologists who investigate the way how semantic information might be represented in the human brain, scientists in the field of computer vision have their own difficulties in finding an appropriate abstract representation and efficient application of top-down data.

My work aims at making a step towards efficient image understanding through the possibilities of fusing the top-down and the bottom-up approaches. In this dissertation I put the focus on the investigation of bottom-up image segmentation and the points where injection of top-down knowledge into a data-driven segmenter is possible.

The first problem I have been working on was the fast segmentation of high resolution images. The motivation for using high resolution images is that this way one can obtain more information from the segmentation output, with objects or object details that otherwise could not be retrieved due to their small extent. Identification of a higher number of details can enhance the robustness of recognition and classification and it can also provide additional cues that top-down knowledge could be applied to. The downside of increasing the physical resolution, i.e. the number of pixels is that the amount of data to be processed grows, which has a negative impact on the running time. To overcome this problem, I constructed and implemented a framework that works in a data-parallel way, and therefore it can efficiently utilize the powerful computational capabilities of many-core environments. The principal idea of the system was inspired by the mean shift algorithm [1], which I extended with a quasi-random sampling scheme for further acceleration, and a cluster merging procedure to reduce

DOI:10.15774/PPKE.ITK.2013.005

Figure 1.1: An intuitive example for differences in content amount. Both images have a resolution of 14.5 megapixels, but while the one on the left contains a single object in front of a homogeneous background, the image on the right has far more details.

over-segmentation. In addition, I proposed a method (named abridging) to reduce the overhead caused by parallelization.

The second problem I addressed was making an adaptive image sampling scheme, in order to take the local content of the image to be segmented into account. I had two simultaneous goals. On one hand, to eliminate superfluous computations for homogeneous regions with minimal content, thus to keep the amount of time required for the segmentation to a content-dependent minimum. On the other hand, to preserve an output quality similar to that of the naïve version, which is applied to every single input pixel without using any kind of sampling. To achieve this, I need not only to determine the number of samples required to maintain a certain segmentation quality, I also have to face the difficulty of finding a good spatial distribution of the samples. I developed an automated mechanism that is capable of solving this task. Additionally, the scheme uses a single-parameter for both the selection of sample candidates and for the registration of the strength of the bond between the pixels and their clusters. This way, the representation of the system remains compact, which enables the segmentation of large images as well. Furthermore, this bond confidence strategy enables each pixel to be associated with the most similar class, with respect to its spatial position and color. I designed the adaptive sampling method to fit into the realized parallel framework.

Observe Figure 1.1 as an illustration for the intuitive justification of the content-adaptive concept.

Quantitatively, both images in this figure consist of more than 14 million pixels.

DOI:10.15774/PPKE.ITK.2013.005

Theoretically, for a naïve, data-driven segmentation algorithm it would take about the same amount of time to segment them, since it considers each and every pixel in the same manner. Applying a sampling technique can definitely accelerate the procedure, but it is not difficult to see that if we aim at maintaining the same level of detailedness in the two segmented images, some regions of the image on the right will be required to be sampled a lot more dense than in the case of the left one.

During the analysis of a huge variety of generic, real-life images of different resolutions, I realized that the inputs are likely to have huge differences in the amount of content, and that the majority of them shows remarkable redundancy in the feature characteristics that could be exploited for the acceleration of the segmentation procedure. Thus, the need for content adaptivity was established, as this way, inhomogeneous image regions containing many details could be sampled densely, thus such rich information is kept in the output, while homogeneous regions may be sampled loosely, such that the segmentation of these regions could be fast.

I also observed that besides "traditional", well-known analytical aspects (such as running time, parallel scalability, or output accuracy that can be measured by various metrics), the amount the of content in the input image should be taken into account not just for faster and more efficient segmentation, but for more accurate evaluation and comprehensive comparison of such lossy segmentation algorithms as well. The amount of processing (thus: the running time) of these techniques can show a greater variance than the non-sampling methods, which might make running time comparison difficult. More importantly, as it will be discussed in Subsection 2.5.3, human-made ground truth provided as the reference for the evaluation of output quality is difficult to accurately supply even for images of low resolutions and it often incorporates subjective factors. Consequently, the comparison of the output quality of different algorithms is not straightforward.

In the computer vision community, characterizing content from the information theoretical point of view is not a novel concept, however, proper description and quantification of the image content is still under ongoing research, and therefore it is not covered by the present dissertation beyond a certain depth. On the other hand, to cope with the lack of a numerical metric, I invented a subjective measure (the kappa-index) that is assigned by humans to describe the complexity of image content. Besides

DOI:10.15774/PPKE.ITK.2013.005

the evaluation made utilizing many well-known and widely used metrics, the correlation analysis of the running times of the proposed framework and the kappa-index has proven that the proposed adaptive system can segment images of less content faster but at the same time it preserves the details of busy image regions.

As of today, if I could start all over again from the beginning, I would definitely switch the order, and pursue adaptivity first before parallelism, because posteriorly it seems more rational to construct the efficient sampler first, and speed it up even more by making it parallel only after[1]. However, the investigation of the second problem actually arose from the findings of the evaluation of the parallel framework that was constructed prior to the content-adaptive scheme. For this reason, by the time the algorithmic background of the content-adaptive extension was ready and was mapped into the parallel scheme, neither the hardware, nor the datasets used for the evaluation of the parallel system were the same.

Consequently, the framework of the dissertation follows the sequence of my research.

Chapter 2 gives a short introduction to image segmentation and the problems and concepts lying in this field. The discussion is started by considering the fundamentals of the two main design approaches: the top-down approach and the bottom-up approach. Covering their pros and cons, the focus is put on the bottom-up scheme, as it is employed in most segmentation systems that are constructed regardless of whether they are built upon top-down or even top-down-bottom-up hybrid inner structures. The most prominent bottom-up algorithms are summarized and briefly evaluated. The mean shift nonparametric segmentation algorithm is discussed in depth, because it was used as the basis of the proposed framework. The end of the chapter summarizes the main properties of publicly available datasets and the supplied metrics that are nowadays the most widely used tools for the evaluation and comparison of segmentation algorithms. Since to my knowledge there exist no high resolution databases for such purposes, the image sets used to assess the framework are described. It is argued that the evaluation made in the high resolution domain should be enhanced with the analytical aspect of image content, thus a subjective rating is defined to characterize it.

[1]Although this sequence might have introduced constraints during the parallelization of a sampler that was originally designed to be sequential.

DOI:10.15774/PPKE.ITK.2013.005

1. INTRODUCTION

Chapter 3 describes the design of the generic building blocks of the parallel segmentation framework that consists of two phases. With the focus put on parallelism, the first phase decomposes the input by nonparametric clustering. In the second phase, similar classes are joined by a merging algorithm that uses color and adjacency information to obtain consistent image content. The core of the segmentation phase is the mean shift algorithm that was fit into the parallel scheme. In addition, feature space sampling is used as well to reduce computational complexity, and to reach additional speedup. The system was implemented on a many-core GPGPU platform in order to observe the performance gain of the data-parallel construction. The chapter discusses the evaluation made on a public benchmark and the numerical results proving that the system performs well among other data-driven algorithms. Additionally, detailed assessment was done using real-life, high resolution images to confirm that the segmentation speed of the parallel algorithm improves as the number of utilized processors is increased, which indicates the scalability of the scheme.

Chapter 4 discusses the method of how the building blocks of the parallel algorithm were extended to operate with respect to the content of the input image. In case of the segmentation phase, the bond confidence concept is introduced, which incorporates an intelligent sampling scheme and a nonlinear pixel-cluster assignment method. The proposed sampling can adaptively determine the amount and spatial position of the samples based on the local properties of the image and the progress of the segmentation. Sampling is driven by a single bond confidence value that is calculated without overhead during the mean shift iterations. The same parameter guides the pixel-cluster mapping that can ensure that each picture element is associated with a class having the most similar characteristics. The method of determining similarity in the merging phase has been extended to tolerate the rapid changes in intensity, hue, and saturation, which occur frequently in real-life images. The focus during the evaluation of the framework has been put onto output accuracy that is measured on three publicly available datasets using numerous metrics and a high resolution image set. The detailed results underline that the output quality of the framework is comparable to the reference but works an order of magnitude faster.

Chapter 5 contains the summary of the dissertation with a short discussion on the methods of investigation, my theses that encapsulate the new scientific results, and examples for the application of my results.

DOI:10.15774/PPKE.ITK.2013.005

The dissertation is closed with the References and the Appendix containing additional segmentation examples.

DOI:10.15774/PPKE.ITK.2013.005

1. INTRODUCTION

DOI:10.15774/PPKE.ITK.2013.005

Chapter 2

Image Segmentation

This chapter gives a short introduction to image segmentation and to the problems and concepts lying in this field. The discussion starts by considering the fundamentals of the two main design approaches: the top-down approach and the bottom-up approach. Covering their pros and cons, the focus is on the bottom-up scheme, as it is employed in most segmentation that are systems constructed regardless of whether they are built upon top-down or even top-down-bottom-up hybrid inner structures. The most prominent bottom-up algorithms are summarized and briefly evaluated. The mean shift nonparametric segmentation algorithm is discussed in depth, because it was used as the basis of the proposed framework. The end of the chapter summarizes the main properties of publicly available datasets and the supplied metrics that are nowadays the most widely used tools for the evaluation and comparison of segmentation algorithms. Since to my knowledge there exist no high resolution databases for such purposes, the image sets used to assess the framework are describe. It is argue that the evaluation made in the high resolution domain should be enhanced with the analytical aspect of image content, thus a subjective rating is defined to characterize it.

DOI:10.15774/PPKE.ITK.2013.005

2.1 Introduction

By the segmentation of an image we mean the partitioning of its pixels. Segmentation is a broad discipline in the field of image processing and computer vision[2] and is applied as a crucial intermediate step in several different tasks of pattern recognition, detection, and high-level image understanding. As most tasks in these fields are relatively easy for the human observer, algorithms designed for segmentation often try to get inspiration from the biological procedure of human visual perception [3, 4, 5, 6, 7]. The latest computational/algorithmic interpretation of segmentation is modeled as the interactive processing of two streams with opposite direction: the data obtained by top-down (or *knowledge-driven*) analysis and data gained via bottom-up (or *data-driven*) analysis [8]. Bottom-up information stands for the set of attributes acquired directly from the raw input material, and top-down information represents the *a priori*, semantic or acquired knowledge that is embedded in the segmenter. Consequently, generic, multipurpose segmentation frameworks are easier to design utilizing data-driven methods, because they use a finite set of rules in low-level attribute spaces in which both local and global features can be extracted on demand. On the other hand in real-life tasks target objects have diverse appearance and in most cases complex hierarchy, such that they can be better isolated, when additional top-down information is available [9]. *Bridging the semantic gap* [10], as the synthesis of top-down and bottom-up approaches is often referred to, is still under heavy research, as so far no successful attempts have been made to find a representation applicable in both approaches.

The difficulty of the initiative is that compact taxonomies, efficient for top-down methods, are too abstract for bottom-up procedures, whereas pixel based representations are hard to aggregate into useful high level information required by the knowledge-driven direction. Another major difference between the data-driven and knowledge-driven approaches is that while a bottom-up system can be employed by itself, a top-down structure requires the help of cues obtained via a bottom-up analysis [11, 12, 13, 14]. For this reason, state of the art segmentation algorithms either choose to apply area-specific top-down information, thereby restricting themselves to a given segmentation

[2] As Gonzales and Woods [2, Ch. 1] point out, the limits between image processing and computer vision are rather soft as most boundaries are artificial and limiting. Consequently, these two will be used in a similar manner throughout this dissertation.

DOI:10.15774/PPKE.ITK.2013.005

task, or they follow the data-driven scheme and utilize low-level properties with high descriptive power, but with somewhat lower accuracy on the object level [15].

My investigation is centered around the unsupervised subbranch of clustering methods that follow the bottom-up scheme and are applied to generic color images. The motivation behind this choice is that these methods are widely used in practical scenarios due to their autonomous nature, relatively low complexity, and discrete length rule collection.

2.2 Bottom-up Image Segmentation

In this section the basic notions of data-driven segmentation are briefly covered and the strengths and challenges connected to the approach are summarized. The second part of the section gives an overview of the most frequently used image segmentation algorithms that follow the bottom-up scheme.

In the field of image segmentation, *features* are typically characteristic attributes of a single pixel that are either original/provided (such as e.g. color channel intensity or the topographic position in the mesh) or derived (such as edge information or the impulse response of a filter). The *feature space* is formulated via the concatenation of the features, and its dimensionality (and consequently: the feature space representation of a pixel) equals the number of features. In the algorithmic level, pixels are represented in an abstract form by *feature space elements* (*FSEs*), such that $\Gamma : P_I \to \mathcal{F}$ denotes the function from picture element indices $P_I = \{1, ..., n\}$ of the input image I (where $n = |I|$) to the feature space $\mathcal{F}$. Then, $\forall i \in P_I$, $\Gamma(i) = \chi_i \in \mathcal{F}$. This numerical representation puts quantities of different properties into a unified frame, consequently image processing algorithms can utilize generic methods from various fields such as machine learning, data mining, neural networks or combinatorics.

2.2.1 Problem Formulation

In real-life scenes, objects have varying appearance due to changes in lighting, occlusion, scale, viewing angle, color, etc. To cope with the lack of high-level information, data-driven techniques (such as cue combination [7, 16], various types of graph-based segmentations [17, 18, 19], mixture model fitting [20], superpixels [21] or the mean shift algorithm [1]) use various low-level features, and apply different similarity metrics to

DOI:10.15774/PPKE.ITK.2013.005

formulate of perceptually meaningful clusters. Since the processing is conducted at the pixel level, computational complexity of these algorithms is often superlinear or in some cases even quadratical subject to the size of the input (i.e. the number of pixels). In practice the actual size of the input is also an important factor, because that is what influences running time besides algorithmic complexity. The two main components of the input size are the resolution of the image and the number of features assigned to the pixels.

Increasing the number of features (i.e. dimensions of the feature space) can lead to a better output quality, as it can increase the discriminative power of an algorithm, but this direction does not lead to a universal rule of thumb for two reasons. Reason one is the curse of dimensionality [22], when the data becomes sparse due to the extended number of dimensions, such that robust discrimination becomes difficult. Reason two is that handling such a feature space may lead to a heavy memory load with frequent accesses, which influences the running time in a highly nonlinear manner above a certain image size.

The second aspect of complexity is related to the number of pixels (i.e. number of elements in the feature space), since most tasks in computer vision can highly benefit from using images of increased resolution, as a consequence of which the amount of data to be processed will grow.

For this reason, several acceleration techniques have been proposed since the birth of the algorithms mentioned above, with the aim of reaching higher segmentation speeds while maintaining the same quality level. Speedups are either achieved in a lossless way, with algorithmic optimization and parallelization techniques, or in a lossy manner, which in one way or another involves the reduction of processed data. The main difference between these two methodologies is that the approaches belonging to the former category normally do not affect the output quality, whereas the latter ones usually have a negative impact on it. Hence, the extent of the quality loss should be judged with respect to the speedup gained. Despite the lossy processing, most of such acceleration techniques do not give the user any control over the quality of the segmentation output, eroding this way the benefits of the increased resolution.

DOI:10.15774/PPKE.ITK.2013.005

2.2.2 Related Work

In this subsection segmentation methods that build upon the bottom-up segmentation approaches listed above are considered, with the aim of achieving the highest possible speedup while maintaining a reasonably small (if any) quality corruption. An additional example belonging here is the mean shift method, but since it plays an important role in the proposed framework, it will be discussed in the next section.

Cue combination [23] used in the field of segmentation is a relatively young technique having ancestors coming from the field of boundary detection [24, 25]. The latest variant was introduced by Arbeláez *et al.* [16] in 2011, who designed a composite segmentation algorithm consisting of the concatenation of the globalized probability of boundary (gPb), the ultrametric contour map, and the oriented watershed. The method utilizes gradients of intensity, color channels, texture, and oriented energy (the second-derivative of the Gaussian), each at eight orientations and three scales resulting in a sum of 96 intermediate stages. Their optimal composition into a single detector is obtained by using previously trained parameters. Such a vast palette of features enables the algorithm to be one of the most accurate data-driven segmentation techniques available [16]. The price on the other hand is an enormous computational complexity, resulting in a runtime of several minutes for a single image. Catanzaro *et al.* [26] successfully sped up the computation of the gPb by mapping it to a GPGPU architecture. The drawback of the parallel implementation lies in the increased memory demand of the contour detector, which extremely increases the cost of the hardware required.

In 2004, Felzenszwalb and Huttenlocher [18] described an unsupervised *graph-based segmentation* algorithm, where each pixel is assigned to a node. Edges between nodes have weights representing the dissimilarity between the two connected nodes. The procedure carries out pairwise region comparison and performs cuts to find a Minimum Spanning Tree (MST). The novelty given by Felzenszwalb is that the segmentation criterion is adaptively adjusted to the degree of variability in neighboring regions of the image. To improve this, Wassenberg *et al.* [27] designed a graph-cutting heuristic for the calculation of the MST. Parallel computation is enabled by processing the image

DOI:10.15774/PPKE.ITK.2013.005

in tiles that results in minimum spanning trees. The component trees are connected subject to region dissimilarity and hence, a clustered output is obtained. The system works with a performance of over 10MPixel/s on high resolution satellite photos. However, the article does not give any high resolution segmentation example, nor do the authors provide any numerical evaluation for the low resolution examples displayed.

Salah *et al.* [19] consider image clustering as a maximum flow-minimum cut problem, also known as the *graph cut* optimization. The aim of this algorithm is to find the minimum cut in a graph that separates two designated nodes, namely, the source and the target. Segmentation is done via an implicit data transform into a kernel-induced feature space, in which region parameters are constantly updated by fixed point computation. To improve segmentation quality, the procedure computes the deviation of the transformed data from the original input and also a smoothness term for boundary preserving regularization. The paper presents an extensive overview of segmentation quality including grayscale and color images, as well as real and synthetic data. The algorithm reaches excellent scores in most benchmarks, however, in some cases image size normalization was necessary due to unspecified memory-related issues. Further in this field, Strandmark and Kahl [28] addressed the problem of parallelizing the maximum flow-minimum cut problem. This is done by cutting the graph to subgraphs such that they can be processed individually. Subgraph overlaps and dual decomposition constraints are utilized to ensure an optimal global solution, and search trees are reused for faster computation. The algorithm was tested both on a single machine with multiple threads and on multiple machines working on a dedicated task. Test fields include color images, CT and MRI recordings, all processed with over 10 million samples per second, however, parallelization speedups were not in all cases present. The lack of quality indicators does not allow the reader to observe output accuracy.

The *normalized cuts* spectral segmentation technique was published by Shi and Malik [17] in 2000. Being different from graph cuts, it performs graph partitioning instead of the maximum flow-minimum cut optimization problem. Edge weights represent pixel affinities that are calculated using spatial position and image feature differences. Cuts are done by observing the dissimilarity between the observed sets as

DOI:10.15774/PPKE.ITK.2013.005

well as the total similarity within the sets. The algorithm has a few difficulties. First off, the final number of clusters is a user parameter that needs to be estimated. Second, graph partitioning is computationally more complex than the previously described optimization problems. Third, minimizing the normalized cut is NP-complete. Fourth, memory requirements of this technique are quadratical. To overcome the third problem, Shi traced back the cut operations to a regular eigenvalue problem using approximation. As an alternative, Miller and Tolliver [29] proposed spectral rounding and an iterative technique to reweigh the edges of the graph in a manner that it disconnects, then use the eigenvalues and eigenvectors of the reweighed graph to determine new edge weights. Eigenvector information from the prior step is used as a starting point for finding the new eigenvector, thus the algorithm converges in fewer steps. Chen *et al.* [30] aimed at handling the memory bottleneck arising in the case, when the data to be segmented is large. Two concurrent solutions were compared: the sparsification of the similarity matrix achieves compact representation by retaining the nearest neighbors in the matrix, whereas the Nyström approximation technique stores only given rows or columns. To achieve additional speedup, most matrix operations were encapsulated into a parallel scheme finally both approaches were extensively tested for accuracy and speed discussing many particular details. Results indicated that the approximation technique may consume more memory and has a bit worse output quality, but works faster than the sparsification.

Despite its usual role as being only a preprocessor, the *superpixels* method is also discussed due to the latest improvements. The algorithm was originally introduced by Ren and Malik [21] and is technically a variant of the graph cuts. The normalized cuts algorithm is utilized to produce a set of relatively small, quasi-uniform regions. These are adapted to the local structure of the image by optimizing an objective function via random search that is based on simulated annealing subject to the Markov Chain Monte Carlo paradigm. As the procedure requires multiple runs, the segmentation is relatively slow (in the magnitude of several dozens of minutes for a small image) and requires the training of certain parameters. For a more consistent output, Moore *et al.* [31] added a topographic constraint, such that no superpixel could contain any other, also they initialized the algorithm on a regular grid to reduce computational complexity. The algorithm also utilizes pre-computed boundary maps that can

DOI:10.15774/PPKE.ITK.2013.005

heavily affect the output quality. Another fast superpixel variant (called *turbopixels*) was proposed by Levinshtein *et al.* [32], who utilized a computationally efficient, geometric-flow-based level-set algorithm. As a result, the segments had uniform size, adherence to object boundaries, and compactness due to a constraint which also limited under-segmentation. Another variant, called *simple linear iterative clustering* (SLIC) was proposed by Achanta *et al.* [33]. The algorithm is initialized on a regular grid, then cluster centers are perturbed in a local neighborhood, to the lowest gradient position. Next, the best matching pixels from a square neighborhood around the cluster center get assigned to the cluster using a similarity measure based on spatial and color information. Finally, cluster centers and a residual error are recomputed, until the displacement of the center becomes adequately small, and connectivity is enforced by relabeling disjoint segments with the labels of the largest neighboring cluster. The algorithm has been reported to achieve an output quality better than turbopixels at a lower time demand due to its linear computational cost and memory usage. Ren and Reid [34] documented the parallelized version (called GPU SLIC, or gSLIC) that achieved a further speedup of 10-20 times compared to the serial SLIC algorithm, such that it runs with 47.61 frames per second on video stream with VGA resolution.

The main difficulty of *mixture models* used for image segmentation lies in the estimation of the parameters used to build the underlying model. In 2007, Nikou *et al.* [20] described a spatially constrained, hierarchical mixture model for which special smoothness priors were designed with parameters that can be obtained via maximum a posteriori (MAP) estimation. In 2010, further improvements were introduced by the same authors [35]: the projection step present in the standard EM algorithm was eliminated by utilizing a multinomial distribution for the pixel constraints. In both papers extensive measurements were performed to evaluate the speed and the output quality of the algorithms. The proposed enhancements make the algorithm accurate, but computationally expensive, furthermore, the number of clusters remains a user parameter. Yang *et al.* [36] proposed to model texture features of a natural image as a mixture of possibly degenerate distributions. The overall coding length of the data is minimized with respect to an adaptively set distortion threshold. Thus, possible redundancy is minimized and the algorithm can merge the data points into a number of Gaussian-like

DOI:10.15774/PPKE.ITK.2013.005

clusters using lower dimensional structures. Good output quality is verified by several different measurements, however, the running time is measured in the magnitude of minutes.

2.3 Mean Shift Segmentation Algorithm

2.3.1 Motivation

During my research made under the umbrella of the data-driven paradigm, I have spent quite a lot of time studying the capabilities and properties of the algorithms described in the previous section, including the relation between the running time and quality of the output. In my opinion, the best segmentation qualities in this field are achieved by algorithms that, at the cost of increased running time, either utilize a vast arsenal of features [16], or define some kind of cluster hierarchy [37] and try to find an optimal matching.

Systems that work with many features perform well because they solve the clustering problem in a "brute-force" way by collecting as many information about the pixels (and often implicitly about their neighborhood regions) as possible.

Hierarchical algorithms are often composed using multiple scales or iterative synthesis. The presence of cluster hierarchy can offer several benefits. Differences in object scale and texture can be handled well, furthermore, the structure of the output clusters can be quickly and easily reorganized according to the desired output. Such systems are usually based on the over-segmentation of the image that is followed by the merging of select segments. This two-level procedure allows the usage of flexible rules that can adapt to the progress of the joinder.

However, both systems have a notable downside. Calculation, caching, and browsing a large number of features requires a huge processing background and, as discussed in Subsection 2.2.1, frequent and heavy memory access. Especially because of the latter, the segmentation speed of such methods is very slow even for images with moderate resolution.

Algorithms that rely entirely upon the created hierarchy can suffer from finding a proper condition for the selection of the optimal scale, which requires the presence of additional heuristics or even *a-priori* information that is often not directly available.

DOI:10.15774/PPKE.ITK.2013.005

In conclusion, if segmentation speed is also a factor besides the quality of the output, hierarchical algorithms are a better choice for segmentation. Making these observations I have selected the mean shift algorithm to be the basis of the proposed framework for the following reasons:

1. **Nonparametric property:** Unlike k-means-like algorithms [38], the mean shift method does not require the number of output clusters to be defined explicitly (see Subsection 2.3.3).

2. **Efficient texture filtering:** The kernel function utilized by the algorithm performs discontinuity preserving smoothing without adding overhead (see Subsection 2.3.3).

3. **Possibility of parallelization:** The algorithm is built on a highly data-parallel [39] scheme that can be employed by many-core systems (see Subsection 3.2.4).

4. **Modular structure:** The algorithm can be modified to construct a hierarchical cluster map in the background. The scheme starts with an over-segmentation that is succeeded by an iterative merging procedure. This modular scheme offers numerous points where task-dependent low-level rules, or optionally, high-level semantic information can be injected (see Sections 4.1, 4.3 and Subsections 4.2.1 and 4.2.3).

5. **Data reduction:** The algorithm can be modified to utilize sampling, which reduces its complexity (see Subsections 3.2.1 and 4.2.1).

6. **Easy extension:** Optionally, the algorithm can easily be extended to work with additional features if required by the segmentation task (see Subsection 2.3.2).

The following subsections discuss the origins and fundamentals of the mean shift method.

2.3.2 Origins: Kernel Density Estimation

The mean shift image segmentation procedure was introduced by Comaniciu and Meer [1] in 2002, highly building upon the work of Cheng [40] and Fukunaga and Hostetler [41].

DOI:10.15774/PPKE.ITK.2013.005

The origin of the algorithm can be derived from kernel density estimation (KDE), which is a robust tool for the analysis of complex feature spaces. Let the feature space be a d-dimensional Euclidean space that is constructed from the input domain via a mapping. Selection of the adequate feature space can highly depend on the given task, but its proper selection results in the benefit that characteristic features get represented in the form of dense regions. Thus, if we consider the feature space as an empirical probability density function, the dense regions will induce high probability. Such local maxima of the function are called *modes*. Image segmentation using KDE is done via retrieving the position of the modes, and associating a subset of data with them based on local properties of the density function. As a preliminary step towards mode seeking, let $\{\chi_i\}, \forall i \in P_I$ denote a set of feature points in a feature space $\mathcal{F} = \mathbb{R}^d$ with distribution $f(\chi)$. The kernel density estimator of this set can be written as

$$\hat{f}_{h,K}(\chi) = \frac{c_{k,d}}{nh^d} \sum_{i \in P_I} k\left(\left\| \frac{\chi - \chi_i}{h} \right\|^2 \right), \tag{2.1}$$

where $h > 0$ is the bandwidth of the nonnegative, radially symmetric kernels of a function K (such as the Gaussian, or the Epanechnikov) that integrates to one because of the normalization constant $c_{k,d} > 0$, $k(x)$ is the profile of kernel K for $x \geq 0$, and n denotes the number of FSEs. Modes of the density function are a subset of the positions where the gradient of the function is zero. Mean shift is an iterative hill climbing algorithm that steps towards the steepest ascent in each iteration. Also, it is proven to converge into locations where the gradient of the estimate is zero [40], which enables it to find the modes without explicit estimation of the density. By following the transformations given in [1, Sec. 2.1], the density gradient estimator can be written in the form of

$$\hat{\nabla} f_{h,K}(\chi) = \frac{2c_{k,d}}{nh^{d+2}} \left[\sum_{i \in P_I} g\left(\left\| \frac{\chi - \chi_i}{h} \right\|^2 \right) \right] \left[\frac{\sum_{i \in P_I} \chi_i g\left(\left\| \frac{\chi - \chi_i}{h} \right\|^2 \right)}{\sum_{i \in P_I} g\left(\left\| \frac{\chi - \chi_i}{h} \right\|^2 \right)} - \chi \right], \tag{2.2}$$

where $g(x)$ is the profile of kernel $G(\chi) = c_{g,d} g(\|\chi\|^2)$. The second term of this equation represents the difference between the weighted mean of kernel $G(\chi)$ and its centroid, and is called mean shift (see Figure 2.1).

DOI:10.15774/PPKE.ITK.2013.005

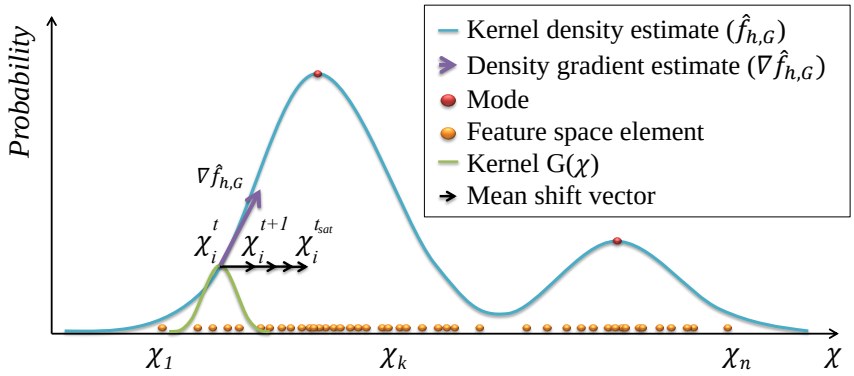


Figure 2.1: Sematic illustration of the mean shift iteration in 1D. Convolving a kernel function with the sparse set of feature space elements (FSEs) gives an estimate for the kernel density. Modes induced by the densest FSE regions encode large, coherent image regions. In each iteration t, the mean of the applied kernel $G(\chi_i^t)$ is calculated, and the kernel steps towards the steepest ascent. Should the distance between previous and current positions of the centroid of the kernel get reasonably small, a mode is found.

2.3.3 Segmenter

Comaniciu utilized the algorithm on a joint feature space consisting of color data (referred to as *range information*) and topographic image coordinates (referred to as *spatial information*). Thus, a feature point is considered to be a five-dimensional vector in the form of $\chi_i = (\mathbf{x}_{r,i}; \mathbf{x}_{s,i}) = (\gamma_{1,i}; \gamma_{2,i}; \gamma_{3,i}; x_i; y_i)$, where $\mathbf{x}_{r,i}$ and $\mathbf{x}_{s,i}$ represent the three-dimensional range coordinates in the selected color space and the spatial coordinates in a two-dimensional mesh of pixel i, respectively. In case the kernel is Gaussian, its property of separability can be exploited and the mean shift vector in the joint feature space can be written in the form of

$$\chi_j^{t+1} = \frac{\sum_{i \in P_I} \chi_j g\left(\left\|\frac{\mathbf{x}_{r,i} - \mathbf{x}_{r,j}^t}{h_r}\right\|^2\right) g\left(\left\|\frac{\mathbf{x}_{s,i} - \mathbf{x}_{s,j}^t}{h_s}\right\|^2\right)}{\sum_{i \in P_I} g\left(\left\|\frac{\mathbf{x}_{r,i} - \mathbf{x}_{r,j}^t}{h_r}\right\|^2\right) g\left(\left\|\frac{\mathbf{x}_{s,i} - \mathbf{x}_{s,j}^t}{h_s}\right\|^2\right)} \tag{2.3}$$

where $\chi_j^{t+1}, j \in P_I$ is the newly calculated position of the mean at iteration $t+1$, P_I is the set of pixels in input image I, $\mathbf{x}_{r,j}^t$ and $\mathbf{x}_{s,j}^t$ are respectively the range and spatial coordinates of the current position of the mean in the feature space, $\mathbf{x}_{r,i}$ and $\mathbf{x}_{s,i}$ are the range and spatial coordinates of the FSEs within the support of the kernel, h_r and

DOI:10.15774/PPKE.ITK.2013.005

h_s are the respective kernel bandwidth parameters, finally, $g(x)$ denotes the Gaussian kernel:

$$g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_0}{\sigma}\right\|^2\right)=\frac{1}{(\sqrt{2\pi\sigma^2})^d}e^{-\frac{\|\mathbf{x}-\mathbf{x}_0\|^2}{2\sigma^2}}. \tag{2.4}$$

The iterative *mean shift procedure* retrieving the local maxima of the probability density function operates the following way.
For $\forall j \in P_I$:

1. Initialize $\chi_j^0 = \chi_j$.

2. For $t > 0$, compute the new mean χ_j^t of kernel j using 2.3, and center the kernel window into this position.

3. If *stopping criterion*

$$\left\|\chi_j^t - \chi_j^{t-1}\right\| < \varepsilon \tag{2.5}$$

   is satisfied for a given threshold ε, then continue to step 4, otherwise go to step 2. (Note: this dissertation refers to the phenomenon of meeting this criterion as *saturation*, for which the time instant is denoted by t_{sat}.)

4. Store the feature space position of χ_j^t into the *output vector* ψ_j.

A subset $B_j = \{\chi_i \in \mathcal{F} : i \in P_I, |B_j| \leq n\}$ that converges into a small tolerance radius of a ψ_j location is the *basin of attraction* of that mode. The FSEs in the basin of attraction belong to its cluster and inherit the color information of mode ψ_j. Sets of mode candidates lying in a close neighborhood are joined together into a single mode. Robustness of a pixel-cluster assignment for a given pixel can be tested by observing the position of the saturation in the case when the mean shift iteration is reinitialized from a slightly perturbed seed point (see the *capture theorem* in [1, Sec. 2.3]). Subsequent to the saturation of all kernels initialized from $\chi_i, \forall i \in P_I$ feature points, the image pixels can be decomposed into $p \ll n$ non-overlapping segments of similar color defined by their respective modes ψ_p. Clusters with an element number smaller than the *smallest significant feature size* M are eliminated.

Figure 2.2 displays the flowchart of the mean shift in the way it was proposed by Comaniciu and Meer.

DOI:10.15774/PPKE.ITK.2013.005

2. IMAGE SEGMENTATION

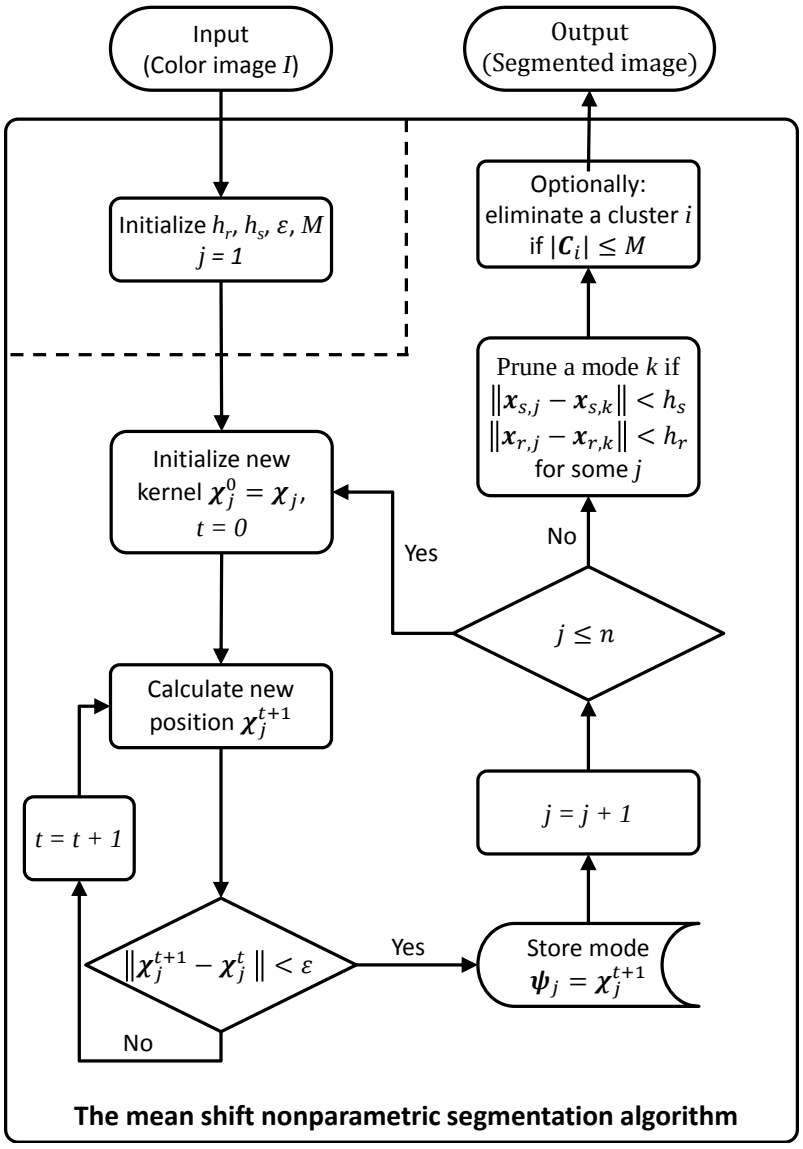


Figure 2.2: Flowchart of the mean shift nonparametric segmentation algorithm. Operation *Feature space transformation* (denoted by Γ) is displayed for the sake of clarity. P_I denotes the pixel indices of input image I, $h_r, h_s, \epsilon, \chi, \psi, C$ and M denote the range and spatial bandwidth, the termination threshold for the mean shift iterations, a feature space element, a mode candidate, a cluster and the smallest significant feature size, respectively. Note that the final elimination step is optional.

DOI:10.15774/PPKE.ITK.2013.005

Despite the listed advantages, the algorithm has a notable downside. Since the naïve version, as described above, is initiated from each element of the feature space, the computational complexity as pointed out by Cheng [40] is $\mathcal{O}(n^2)$ with the main bottlenecks being the calculation of the weighted average and the retrieval of neighboring pixels in the feature space.

2.4 Acceleration Strategies

Several techniques were proposed in the past to speed up the procedure, including various methods for sampling, quantization of the probability density function, parallelization and fast nearest neighbor retrievement among other alternatives. For the sake of a comprehensive overview, the most common and effective types of acceleration available in the literature are arranged into two main groups depending on the way of the approach. These algorithms are discussed in the next two subsections.

2.4.1 Algorithmic Modifications

The first group of methods achieves faster segmentation performance via the modification of the algorithm itself.

DeMenthon *et al.* [42] reach lower complexity by applying an increasing bandwidth for each mean shift iteration. Speedup is achieved by the usage of fast binary tree structures that are efficient in retrieving feature space elements in a large neighborhood, while a segmentation hierarchy can also be built at the same time.

Yang *et al.* [43] accelerate the process of kernel density estimation by applying an improved Gaussian transform, which boosts the summation of Gaussians. Enhanced by a recursively calculated multivariate Taylor expansion and an adaptive space subdivision algorithm, their method reached linear running time for the mean shift. In another paper [44] Yang *et al.* used a quasi-Newton method. In this case, the speedup is achieved by incorporating the curvature information of the density function. Higher convergence rate is realized at the cost of additional memory and a few extra computations.

Georgescu *et al.* [45] speed up the nearest neighbor search via locality sensitive hashing that approximates the adjacent feature space elements around the mean. As the number of neighboring feature space elements is retrieved, the enhanced algorithm

DOI:10.15774/PPKE.ITK.2013.005

can adaptively select the kernel bandwidth, which enables the system to provide a detailed result in dense feature space regions. The performance of the algorithm was evaluated through texture segmentation task as well as through the segmentation of a fifty-dimensional hypercube.

Several other techniques are proposed by **Carreira-Perpiñán** [46] to achieve speedups: he applied neighborhood subsets, spatial discretisation and an algorithm based on expectation-maximization [47]. From these variants, spatial discretisation turned out to be the fastest. This technique divides the spatial domain of the image into cells of subpixel size and forces all points projecting on the same cell to converge to the same mode. This way the total number of iterations is reduced. He also analyzed the suitability of Newton's method, and later on proposed an alternative version of the mean shift using Gaussian blurring [48], which accelerates the rate of convergence.

Luo and Khoshgoftaar [49] use the mean shift [1] to create the over-segmentation of the input. The resulting clusters are then merged utilizing multiscale region merging that is guided by the minimization of a minimum description length–based criterion.

Comaniciu [50] proposed a dynamical bandwidth selection theorem, which reduces the number of iterations till convergence, while at the same time it determines the proper kernel bandwidth to be used. The method estimates the most stable covariance matrix for each data point across different scales. Although the analysis is unsupervised the range of scales at which the structures appear in the data has to be known *a priori*. The selected bandwidth matrices are employed in the variable-bandwidth mean shift for adaptive mode detection and feature space partitioning.

Wang *et al.* [51] utilize a dual-tree methodology. A query tree and a reference tree are built during the procedure, and in an iteration, a pair of nodes chosen from the query tree and the reference tree is compared. If they are similar to each other, a mean value is linearly approximated for all points in the considered node of the reference tree, while also an error bound is calculated. Otherwise the traversal is recursively called for all other possible node pairs until it finds a similar node pair (subject to the error boundary), or reaches the leaves. The result of the comparison is a memory efficient cache of the mean shift values for all query points speeding up the mean shift calculation. Due to the applied error boundary, the system works accurately, however the query tree has to be iteratively remade in each mean shift iteration at the cost of additional computational overhead.

DOI:10.15774/PPKE.ITK.2013.005

Lastly, the work of **Wang** *et al.* [52] is mentioned, who by the use of anisotropic kernels aim at improving the quality rather than the speed of the segmentation procedure. The benefit of these kernels over simple adaptive solutions is that they adapt to the structure of the input data, therefore they are less sensitive to the initial kernel bandwidth selection. However, the improvement in robustness is accompanied by an additional cost of complexity. The algorithm was tested on both images and video, where the 5D feature space was enhanced with a temporal axis.

2.4.2 Feature Space Modifications

The second group of methods focuses on reducing the content of the feature space, so that segmentation can be performed on a smaller amount of data, decreasing the number of required calculation steps.

Guo *et al.* [53] aim at reducing the complexity by using resampling: the feature space is divided into local subsets with equal size, and a modified mean shift iteration strategy is performed on each subset. The cluster centers are updated on a dynamically selected sample set, which is similar to the effect of having kernels with iteratively increasing bandwidth parameter, therefore it speeds up convergence.

Paris and Durand [37] employed a hierarchical segmentation scheme based on the usage of Morse-Smale complexes. They used explicit sampling to build the coarse grid representation of the density function. The separability property of the Gaussian convolution is exploited to quickly extract the modes of the function, then clusters are formulated using a smart labeling solution with simple local rules. The algorithm does not label pixels in the region of cluster boundaries; this is done by an accelerated version of the mean shift method. Additional speedup was obtained by reducing the dimensionality of the feature space via principal component analysis.

Pooransingh *et al.* [54] initialize kernels from randomly sampled positions of the feature space. At each iteration, the center of mass was calculated using the feature space elements situated within the range bandwidth. After convergence, the FSEs involved in the procedure inherited the color information of the found mode. This way, a reduced number of samples is used to cluster the input, thus the computational demand is decreased.

Zhou *et al.* [55] employed the mean shift procedure for volume segmentation. In this case the feature space was tessellated with kernels resulting in a sampling of initial

DOI:10.15774/PPKE.ITK.2013.005

seed points. All mean shift kernels were iterated in parallel and as soon as the position of two means overlapped, they were concatenated subject to the assumption that their subsequent trajectory will be identical. Consequently, complexity was reduced in each iteration giving a further boost to the parallel inner scheme. Sampling on the other hand was performed using a static grid which may result in loss of information in the case when there are many small details on the image.

Xiao and Liu [56] also propose an alternative scheme for the reduction of the feature space. The key element of this technique is based on the usage of kd-trees. The first step of the method is the construction of a Gaussian kd-tree. This is a recursive procedure that considers the feature space as a d-dimensional hypercube, and in each iteration splits it along the upcoming axis in a circular manner until a stopping criterion is met, providing a binary tree. In the second step of this algorithm, the mean shift procedure is initialized from only these representative leaf elements resulting in modes. Finally, the content of the original feature space is mapped back to these modes. The advantage of this sampling scheme is decreased complexity, which, along with the utilization of a GPGPU, boosted the segmentation performance remarkably.

Freedman and Kisilev [57, 58] apply sampling on the density function, forming an approximated version of the kernel density estimate. The mean shift algorithm is initialized from every sample of the compact KDE, finally each element of the original data set is mapped backwards to the closest mode obtained with the mean shift iteration.

Zhang *et al.* [59] approached the problem of complexity from the aspect of simplifying the mixture model behind the density function, which is done using function approximation. As the first step, similar elements are clustered together, and clustering is then refined by utilizing an intra-cluster quantization error measure. Simplification of the original model is then performed using an error bound being permanently monitored. Thus the mean shift run on the simplified model gives results comparable in quality to the variable bandwidth mean shift utilized on the original model, but at a much lower complexity and hence with a lower computational demand.

Finally, the **EDISON system** [60] is considered that is a popular tool for the evaluation of mean shift due to its public availability and straightforward usability. This application implements the mean shift segmentation algorithm as published by Comaniciu and Meer [1], and operates in the Luv color space. Optionally, the EDISON

DOI:10.15774/PPKE.ITK.2013.005

offers speedup strategies present both during the mean shift iterations (using a path assigned strategy) and during the subsequent mode merging (using region adjacency graphs and graph contraction). The system works fast due to its advanced C++ implementation.

2.5 Evaluation

Arguably, the two most frequently used performance indicators utilized to characterize the efficiency of a clustering algorithm are the **running time demand** and the **output accuracy** of the segmenter. Although there exist special cases in which algorithm assessment is driven exclusively by a single aspect, but in general, a comprehensive evaluation needs to incorporate both dimensions. The reason neither of these properties can adequately describe the capabilities of a segmenter *per se* is that in spite of being perpendicular axes of evaluation, there is a strong trade-off between them. Consequently, the task of algorithm assessment is to estimate an optimum along this tradeoff curve with respect to possible priorities. The next subsection highlights some of the fundamental relations of possible aspects of analysis, furthermore the complexity of providing a proper quality description is discussed.

2.5.1 Traditional Analytical Aspects

Being a primary property, it is easy to define and to measure the *running time demand* of a segmentation algorithm: it is the amount of time required to provide the clustered output from the input.

Output accuracy is a much more ambiguous property. The first difficulty we face if we try to characterize accuracy is the methodology of evaluation, for which Zhang *et al.* [61] give the following taxonomy: (i) subjective or objective, (ii) system-level or direct, (iii) analytical or empirical, (iv) supervised or unsupervised. The second main difficulty, as mentioned in the introduction, is the characterization of meaningful segments. Figure 2.3 illustrates these problems through examples.

The parts highlighted in the left side of the figure illustrate the difficulty of finding universal rules of similarity. In real-life images, different objects often have similar color, shape and texture (as shown by region of interests (ROIs) *a_1)* and *a_2)*), but at the same time, intra-object properties can vary significantly (see ROI *b)*). Using the

DOI:10.15774/PPKE.ITK.2013.005

2. IMAGE SEGMENTATION

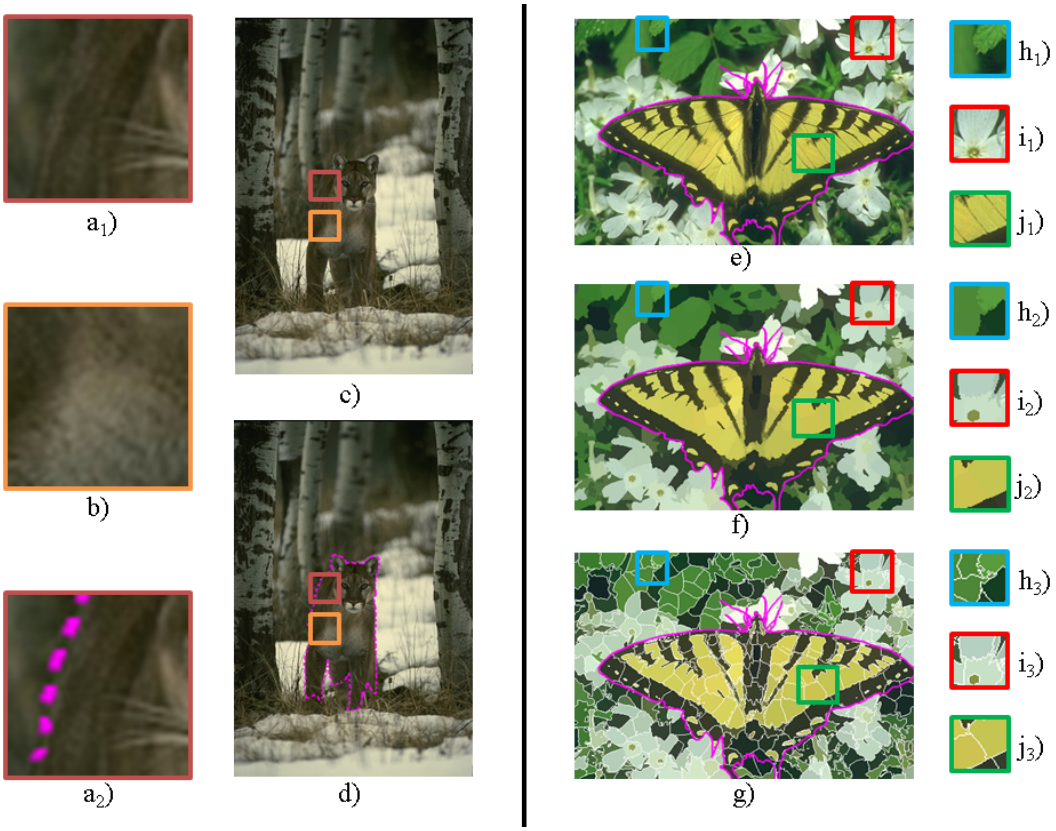


Figure 2.3: Examples for different clustering problems in bottom-up image segmentation. The boundary colors of the region of interest (ROI) windows encode the area in the corresponding image they are taken from. The butterfly and the cougar were delineated manually with purple for the sake of visibility. The left side of the figure illustrates the difficulty of the integration of segments based on static thresholds: pixel regions with similar texture and color properties can belong to different objects (ROIs *a1)* and *a2)*), on the other hand, regions with different colors can belong to the same object (ROI window *b)*). In the right side of the figure an over-segmented version of input image *e)* is shown, with white segment boundary labels (*g)*) and without them (*f)*). Note that for the sake of clarity, the main object has been manually delineated in *e)*. It is quite clear that the clusters covering the flower in ROIs *i2)* and *i3)* should be merged, because they have similar color that is different from their neighborhood. However, in the case of ROIs *h2)* and *h3)* the boundaries of the underlying objects are somewhat ambiguous, because they depend on whether distinguishing the leaves is necessary in the given context, or is it preferable to treat them as a single background cluster. Despite having different colors, the yellow and black parts of the wing present in ROIs *j2)* and *j3)* should also be merged, since they form a single object. (Images are from [62], ROI sizes are 40x40-45x45 pixels.)

DOI:10.15774/PPKE.ITK.2013.005

feature arsenal referred above can work extremely well for these types of problems. The right side of the figure shows an input image *e)*, its over-segmented version with white segment boundary labels *g)* and without them *f)*. Three different cluster merging situations are shown here.

1. Easy task: Intuitively, in ROIs i_2*)* and i_3*)*, the clusters with the similarly grayish color should be merged.

2. Ambiguous task: In ROIs h_2*)* and h_3*)*, the green clusters can be merged depending on the task. In case we are interested in the details of the image background, the leaves should remain independent, otherwise they should be assigned to a single background cluster.

3. Hard task: Clusters in ROIs j_2*)* and j_3*)*, have completely different characteristics, however, they belong to the same object and therefore should be put into the same cluster.

Image segmentation in general is an ill-posed problem in the sense that a *meaningful segment* [63] is determined by the actual task [7, 62, 64, 65] and object boundaries can be highly subjective (see row 5 in Figures 4.6 and B.1 for practical examples).

The third difficulty is the selection of proper metrics for measuring accuracy, which is not straightforward either [66, 67, 68]. Finally, obtained results should be comparable to the results of other algorithms.

As of today, the *de facto* standard for comparing segmentation algorithms is to measure performance on public databases that offer an off-the-shelf solution in a unified framework for the four problems enumerated above. In addition to the Berkeley Segmentation Dataset and Benchmark (BSDS) [16], which is the most widely used dataset, evaluation was also performed on the Weizmann Institute Segmentation Evaluation Database (WIDB) [69]. The results for both datasets will be discussed in detail in Subsection 2.5.3.

While these datasets (especially the BSDS) provide solid background for segmentation quality evaluation, their applicability is somewhat limited, because they contain images of relatively low **resolution**, which can be a third axis of evaluation. In the context of segmentation, resolution is basically equivalent to the number of input image pixels.

DOI:10.15774/PPKE.ITK.2013.005

My argument here is that results measured on the datasets mentioned above can not be generalized onto images of higher resolution for three reasons.

First, as it was discussed in Subsection 2.2.2, most segmentation algorithms have a nonlinear complexity. However, the expected running time demand can be estimated with some precision with advanced complexity analysis.

Second, many algorithms utilize lossy speedup techniques to reduce running times which may introduce an image dependent effect on the running time demand, and even more importantly, on the quality of the output.

Third, fixing the resolution of the input images is in itself insufficient, because it does not say anything about the amount of *useful information* present in the image. As the proposed system is built upon a sampling scheme, it important to address **image content** [64] besides physical resolution. Due to the subjective nature of usefulness of information, it is hard to give a precise definition, but the output complexity of segmentation made by humans can be regarded as a good measure. Since image size and computational demand are highly related, using a higher resolution is only justified if it provides additional useful information, subject to the given task. As lossy algorithms achieve speedups via reduction of the amount of data to be processed, it is their responsibility to neglect only redundant information. If a lossy algorithm acts carelessly in this regard, then the enlargement of the resolution can become of no effect, because the relevant additional details may get lost. In the computer vision community, characterizing content from the information theoretical point of view is not a novel concept, but proper description and quantification of the image content is still under ongoing research. The main goal is to find an adequate formalization of what is considered useful in a human perception aspect. A few approaches that deal with similar tasks involving saliency, entropy, local contrast scale, and other low level descriptors are briefly summarized in the following.

2.5.2 Content—An Additional Analytical Aspect

Image content appears in the literature mostly related to compression, content-based image retrieval (CBIR), and image matching, all having somewhat different aims.

In the case of CBIR and matching, the majority of algorithms focus on finding characteristic regions in the image, and calculate certain similarity measures only for

DOI:10.15774/PPKE.ITK.2013.005

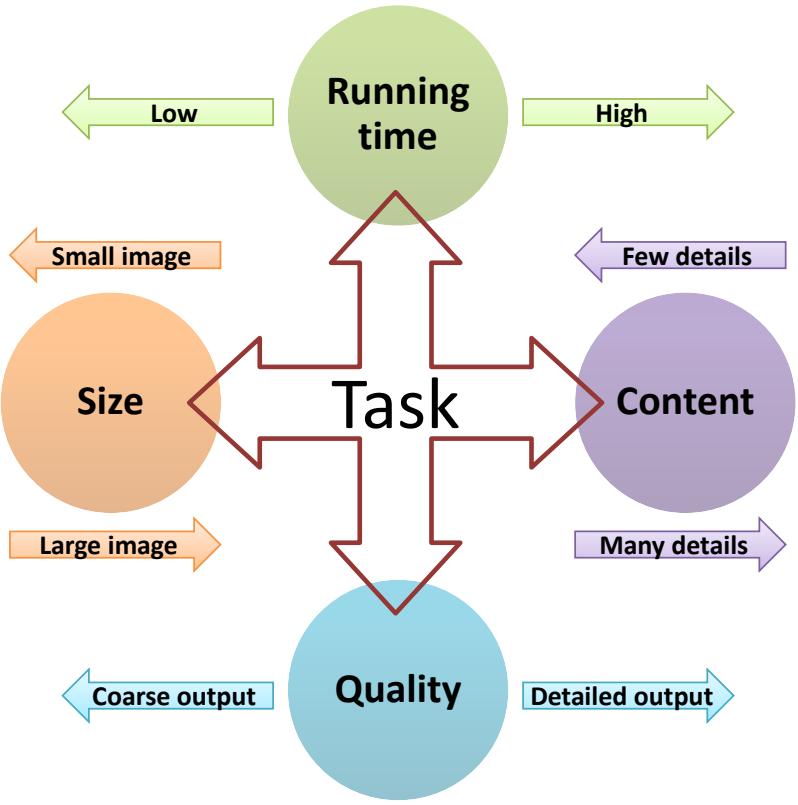


Figure 2.4: The four main aspects required for an extensive performance evaluation of a segmentation algorithm. The significance of each of these aspects is influenced by the goals and priorities of the given task, which determines the optimal tradeoff between them.

the identified region(s) of interest. On the other hand, compression methods work on the entire image, but the aim in this case is to encode it with as few bits as possible.

Kadir and Brady [64] proposed a biology-inspired algorithm aiming at the description of image content in salient image regions. According to this approach, saliency detection is done in a multiscale environment. For each scale, entropy is calculated from the distribution of local grey-value invariants taken around the detected salient points of interest. The method then searches across the isotropic scale space for scale localized regions with high entropy, which then are used for object recognition, tracking and classification.

Another approach of measuring regional entropy was presented by Yanai and Barnard [70] that estimates the *visualness* of concepts. The paper describes an al-

DOI:10.15774/PPKE.ITK.2013.005

gorithm that performs probabilistic region selection for regions that can be linked with a given concept "X" (e.g. yellow, ancient or female) from images which are labeled as "X" or "non-X". Next, a measure of the entropy of the selected regions based on a Gaussian mixture model for regions is computed. A low level of entropy means that the concept in question can be linked with the region features, but if the entropy is more like that of random regions, then the concept has some other meaning which is not captured by the selected features.

Moghaddam *et al.* [71] uses a purely bottom-up approach to describe image content. Descriptors include color, edge strength and orientation, and texture measurements. Regions of interest were selected online by the user. Main target applications include CBIR and medical applications.

Abbademi *et al.* [72] discusses perceptual texture features, such as coarseness, contrast, direction, and busyness (also they mention regularity and roughness). They conducted psychological experiments to determine the correspondence between the rankings obtained with these computational measures and rankings given by human subjects. All measures showed a strong correlation between the psychological the computational variants with coarseness having the highest correspondence.

In this dissertation no metric is given that could be used for the estimation of the useful content amount in images, rather, a subjective rating to characterize the amount of content in an image is defined. The *kappa-index* (κ), a human perception-based degree that is calculated as the mean of ratings provided by subjects, who are asked to assess the amount of useful content in the image. A scale from 1 to 5 have been used, where 1 means a *"sparse image that contains only a few objects and large, homogenous regions"*, and 5 refers to a *"packed image having many identifiable details and rich information content"*. For a given image I, the *kappa-index* is obtained the following way:

$$\kappa(I) = \frac{\sum_{u \in U} \mathsf{r}_u(I)}{|U|} \tag{2.6}$$

where I denotes the input image, U is the set of participants and $\mathsf{r}_u(I)$ is the rating assigned to image I by participant u.

Since the algorithm described in Chapter 4 uses a content-adaptive sampling scheme, a high correlation was expected to be observed between its running time and the kappa-index.

DOI:10.15774/PPKE.ITK.2013.005

Finally, although this dissertation does not address it, **energy consumption** is mentioned for the sake of completeness, as it is also a frequently used aspect of evaluating algorithmic efficiency.

2.5.3 Databases

Due to its public availability, the standardized evaluation framework provided with it, and the versatile, well-documented metrics it uses, the **Berkeley Segmentation Dataset and Benchmark** can be considered a quasi-standard tool for the evaluation of segmentation and boundary detection algorithms. Its first version (referred to as BSDS300 [62]) contains 200 *training images* and 100 *test images* of natural scenes. The current version (referred to as BSDS500 [16]) utilizes all images of the BSDS300 as training images and has an additional test set of 200 images. Since the benchmark provided for the current version is backwards compatible, output accuracy is still worth measuring on the BSDS300 as well, as it is comparable with existing results. Both datasets provide multiple human-labeled segmentations for each image as ground truth, thus output quality assessment can somewhat take into account the ambiguity caused by the segmentation task (as discussed under "output accuracy" in Subsection 2.5.1), hence a perfect segmentation score is practically impossible. The evaluation method follows the traditional validation scheme on both datasets, such that algorithm development and parameter optimization is allowed only on the training set. Once an optimal system setting is found, quality assessment in done on the test set only with this setting. The resolution of all images is 481×321 pixels. Albeit being a popular assessment tool, the BSDS does not stand without criticism. As the authors note as well, images in the dataset offer a limited range of scales due to their static resolution and photographic bias. The consequence, as Alpert *et al.* [69] pointed out, is that the delineation of objects present in the references are often subject to semantic considerations, moreover, ground truth images are often under-segmented.

Alternatively, they proposed the **Weizmann Institute Segmentation Evaluation Database** (WIDB) [73] that takes a different approach to assess the output quality of clustering algorithms. The WIDB consists of two sets of grayscale images, each containing 100 samples. The difference between the sets is that the images in the first one contain one foreground object that stands out from its surroundings (in terms of grayscale intensity or texture), whereas the second set has images with two

DOI:10.15774/PPKE.ITK.2013.005

individual foreground objects that often have different scales as well. Multiple human-made ground truth segmentations are available for all images, but unlike in the case of the BSDS, participants preparing the reference were asked to mark the foreground object(s) only. A pixel is considered to belong to a foreground object only if it was marked so by at least two annotators.

In this regard, the WIDB is rather an evaluation set for foreground-background separation than generic segmentation. Each image is 300 pixels wide with various heights.

As for high resolution images, I have no knowledge of a publicly acknowledged, annotated dataset that is suitable for such evaluation tasks. As it was discussed in the previous subsection, image and reference image interpolation is not viable. Such a framework should consider novel aspects that were not required to be taken into account in the lower resolution case, simply because of image scale.

First off, taking into account image content is not circumventable (see Figure 1.1). Furthermore, high resolution images may contain more complex information that is often further strained with environmental "noise", such as shading or reflection. In the case of adaptive and other lossy segmentation methods this fact makes it inevitable to make a deliberate positioning along the task-dependent tradeoff curve between output quality and runtime, since deviations in running time are directly related to the amount of content in the image. Content-based metrics could also provide information about the descriptive and discriminative power of possible features and cues.

Second, the framework should not punish over-segmentation as much as it currently does for the relatively small images, since the original aim of using high resolution inputs is to gain more detailed information.

Third, since high resolution images can contain a huge amount of information, the traditional unsupervised scheme using human-made ground truth may lead much greater variations in the references, which might require either more advanced comparison metrics (see Subsection 2.5.1), or a fundamentally different evaluation methodology.

Further aspects and an overview of existing unsupervised evaluation alternatives are discussed in detail by Zhang *et al.* [61]. The composition of such a niche framework exceeds the scope of the current dissertation, but it would be very important for the standardized evaluation of segmentation techniques on high resolution images.

DOI:10.15774/PPKE.ITK.2013.005

The primary aim of **high resolution measurements** was, to assess the running time of the segmentation algorithm. However, the secondary aspect during the evaluation of the parallel system (discussed Chapter 3) and the adaptive framework (discussed in 4) was different.

Since in the former case, such measurements were meant to demonstrate how the system performs on an enhanced number of FSEs, an image set consisting of 15 high quality images was formulated in five different resolutions.

In the latter case, the main dimension of evaluation was not how the alternation of resolution influences the running time, but how the varying amount of content does. Consequently, a test set consisting of 103 color images have been complied, each having a resolution of 10 megapixels. Next, 15 subjects were asked to rate the useful content of all images and the kappa-index was calculated using Equation 2.6.

2.5.4 Metrics

The definition and the main characteristics of the standard metrics provided by the datasets are briefly summarized in the following. In the definitions it is assumed that O is the output segmentation to be evaluated, and $\mathbb{G}$ is a set of ground truth segmentations.

1. Segmentation Covering

The *Segmentation Covering* ($\mathcal{C}$) (not to be confused with either FSEs that a kernel might "cover", or coverage, the percentage of bound FSEs) is a region-based metric that measures the normalized average of the topographic overlap between clusters of the segmentation output and the ground truth clusters. Let C and C' denote a ground truth segmentation $G \in \mathbb{G}$ and regions (clusters) of O respectively. Their *normalized overlap score* ω is then defined as

$$\omega(C, C') = \frac{|C \cap C'|}{|C \cup C'|}, \tag{2.7}$$

where $\omega \in [0, 1]$. Then, the covering of the output segmentation by a ground truth segmentation is

$$\mathcal{C}(O, G) = \frac{1}{n} \sum_{C' \in O} |C| \max_{C \in G} \omega(C, C'), \tag{2.8}$$

DOI:10.15774/PPKE.ITK.2013.005

where n is the total number of pixels in the image. Consequently, the covering $\mathcal{C}(O, \mathbb{G})$ is defined as the average of covering values of all ground truth segmentations in $\mathbb{G}$. The definition implies that the domain of $\mathcal{C}$ is $[0, 1]$, and higher values represent segmentations closer to the ground truth(s). The metric is supplied for the BSDS500 framework.

2. Probabilistic Rand Index

The *Probabilistic Rand Index* (PRI) is a pixel-based metric that compares pixel pairs in the segmentation output to ground truth segmentations:

$$PRI(O, \mathbb{G}) = \frac{1}{\binom{n}{2}} \sum_{\substack{i,j \\ i<j}} [b_{ij}(O)p_{ij}(\mathbb{G}) + (1 - b_{ij}(O))(1 - p_{ij}(\mathbb{G}))], \tag{2.9}$$

where

$$b_{ij}(Q) = \begin{cases} 1, & \text{if } \mathfrak{C}_i^Q = \mathfrak{C}_j^Q, \\ 0, & \text{else} \end{cases} \tag{2.10}$$

is a binary number denoting whether a pair of output image map pixels i and j have the same pixel-cluster mapping values (see Section 4.1) in segmentation Q, and the probabilities p_{ij} are estimated by averaging b_{ij} over the ground truth set:

$$p_{ij}(\mathbb{G}) = \frac{\sum_{G \in \mathbb{G}} Pr(b_{ij}(G) = 1)}{|\mathbb{G}|}. \tag{2.11}$$

The definition implies that the domain of PRI is $[0, 1]$, and higher values represent segmentations closer to the ground truth(s). The metric is supplied for the BSDS500 framework.

3. Variation of Information

The *Variation of Information* (VI) is an entropy-based metric. For output image O and ground truth $G \in \mathbb{G}$ it is calculated as:

$$VI(O, G) = H(O) + H(G) - 2\mathcal{I}(O, G) \tag{2.12}$$

where H denotes the entropy of a segmentation Q:

$$H(Q) = -\sum_{C' \in Q} \frac{|C|}{n} \log \frac{|C'|}{n}, \tag{2.13}$$

DOI:10.15774/PPKE.ITK.2013.005

and $\mathcal{I}$ is the mutual information between two segmentations:

$$\mathcal{I}(Q, Q') = \sum_{C \in Q} \sum_{C' \in Q'} \frac{|C \cap C'|}{n} \log \left[n \frac{|C \cap C'|}{|C||C'|} \right]. \tag{2.14}$$

The definition implies that the domain of VI is $[0, \infty)$, and lower values represent segmentations closer to the ground truth. The perceptual meaning and applicability of this measure is unknown when more than one ground truth images are given [16]. The metric is supplied for the BSDS500 framework.

4. F-Measure

The *F-measure* (F) is a boundary-based metric used for boundary evaluation. It is given by the harmonic mean of precision and recall:

$$F = \frac{2\mathcal{P}\mathcal{R}}{\mathcal{P} + \mathcal{R}} \tag{2.15}$$

where $\mathcal{P} = {}^{\mathrm{tp}}/_{(\mathrm{tp+fp})}$ denotes the *precision* and $\mathcal{R} = {}^{\mathrm{tp}}/_{(\mathrm{tp+fn})}$ denotes the *recall* with tp, fp and fn standing for *true positive, false positive* and *false negative* hits respectively. The definition implies that the domain of F is $[0, 1]$, and higher values represent segmentations closer to the ground truth(s). The metric is supplied for both the BSDS300, BSDS500 and WIDB frameworks.

5. Average Precision

The *Average Precision* (AP) is a boundary-based metric used for boundary evaluation. It is calculated as $AP = \int_0^1 \mathcal{P}(\mathcal{R}) d\mathcal{R}$, i.e. by plotting the precision as a function of the recall, the average precision is the average value of $\mathcal{P}(\mathcal{R})$ in the interval of $\mathcal{R} \in [0, 1]$. In a discrete system, this integral is written as the following sum:

$$AP = \sum_{k=1}^{|\mathrm{tp+fp}|} \mathcal{P}(k)(\mathcal{R}(k) - \mathcal{R}(k-1)), \tag{2.16}$$

where $\mathcal{R}(k)$ values are in increasing order and $\mathcal{R}(0) = 0$. The definition implies that the domain of AP is $[0, 1]$, and higher values represent segmentations closer to the ground truth(s). The metric is supplied for the BSDS500 framework.

6. Fragmentation

DOI:10.15774/PPKE.ITK.2013.005

The *Fragmentation* (ξ) is a region-based metric proposed by [69] that displays the number of segments used to cover a single foreground object. The definition implies that the domain of ξ is $[0, \infty)$, and lower values represent segmentations closer to the ground truth. The metric is supplied for the WIDB framework.

The values of the metrics are documented along different dimensions in the BSDS500 and in the WIDB (the BSDS300 discusses F-measure in an optimal dataset scale basis that is discussed in the following). The presented values are:

1. **Optimal Dataset Scale (ODS)** (only for metrics $\mathcal{C}, F, PRI$ and VI): the best output quality of the metric obtained on the training set using fixed parametrization/scale for the entire set;

2. **Optimal Image Scale (OIS)** (only for metrics $\mathcal{C}, F, PRI$ and VI): the best output quality of the metric obtained on the training set using fixed parametrization/scale for individual images;

3. **Best Covering (Best)** (only for metric $\mathcal{C}$): the best output quality of the metric obtained on the training set using any possible cross-parametrizations.

2.5.5 Comparison

Giving an extensive comparison of the proposed algorithm with other segmentation methods (see Subsection 2.2.2 and Section 2.4) is very difficult. The main problem is that the majority of these systems was assessed in different environments. That is, not only the input images were often hand picked, but the metrics, the parametrization (if documented), and the hardware used show a huge diversity as well. As a consequence, the results published for these methods are not directly comparable. Such results could only be obtained by reimplementing and reassessing each method using identical, standardized evaluation characteristics and constraints, which exceeds the bounds of this dissertation due to the massive amount of work required. The published properties of the evaluation environments and the best running times and/or acceleration results reported are summarized in Table 2.1.

However, as more and more algorithms are assessed using the unified methodologies introduced by the formerly discussed evaluation databases, the published results

DOI:10.15774/PPKE.ITK.2013.005

Table 2.1: Main details and running time/acceleration results of the mean shift variants as provided by their respective authors. Additional (+) and missing (–) features respected to the most common, five dimensional (spatial-range) feature space are indicated. Acceleration values are relative to the mean shift algorithm in [1]. Parameters denoted by "n/a" are not reported.

Author	Hardware	Resolution/ No. FSEs	Platform	Feature space dimensions	Running time/ (acceleration)
DeMenthon *et al.* [42]	n/a	88×60	C, MATLAB interface	7D (+motion angle)	15 sec
Yang *et al.* [43]	Pentium III 900MHz	481×321	C++, MATLAB interface	3D (–spatial)	12.359 sec
Yang *et al.* [44]	n/a	481×321	n/a	3D (–spatial)	n/a
Georgescu *et al.* [45]	n/a	65536	n/a	48D (hypercube)	276 sec (21.3x)
Carreira-Perpiñán [46]	n/a	100×100	n/a	3D (grayscale)	10-100x
Carreira-Perpiñán [48]	n/a	137×110	n/a	5D	4.8x
Carreira-Perpiñán [47]	n/a	110×73	n/a	3D (grayscale)	2.2x
Luo and Khoshgoftaar [49]	Pentium 4 2.8GHz	481×321	n/a	3D (grayscale)	1.054 sec
Comaniciu [50]	n/a	500×333	n/a	3D (–spatial)	n/a
Wang *et al.* [51]	n/a	n/a	n/a	6D (+time in video)	n/a
Guo *et al.* [53]	Pentium 4 2.93GHz	512×484	C++	3D (–spatial)	2.23 sec (192.12x)
Paris and Durand [37]	AMD Opteron 2.6GHz	3424×2283	C++	5D	14.83 sec
Pooransingh *et al.* [54]	n/a	1000	MATLAB	2D (synthesized)	10-100x
Zhou *et al.* [55]	GeForce 8800GTX	$256 \times 256 \times 256$	n/a	4D (3D spatial and grayscale)	5.3 sec (226.45x)
Xiao and Liu [56]	Pentium E5200 2.5GHz	2256×3008	C++	5D	5.91 sec
Freedman and Kisilev [57, 58]	Pentium C2D 2.53GHz	7,000,000	n/a	5D	5.67 sec
Zhang *et al.* [59]	n/a	512×512	n/a	3D (–spatial)	3.67 sec (910.9x)

DOI:10.15774/PPKE.ITK.2013.005

recently started to become directly comparable. Such results are collected and displayed in Subsection 4.5.1.

For these reasons, the assessment of the proposed algorithm presented in Section 2.4 does not contain results for all the discussed variants of the mean shift, but only for the methods that have been evaluated along the scheme proposed by the authors of the public datasets.

For measuring properties and metrics exceeding the capabilities offered by frameworks provided along with public datasets, comparison with respect to the analytical aspects discussed above is possible using a publicly available reference system, such as EDISON.

DOI:10.15774/PPKE.ITK.2013.005

Chapter 3

Parallel Framework

This chapter describes the design of the generic building blocks of the parallel segmentation framework that consists of two phases. With the focus put on parallelism, first phase decomposes the input by nonparametric clustering. In the second phase, similar classes are joined by a merging algorithm that uses color and adjacency information to obtain consistent image content. The core of the segmentation phase is the mean shift algorithm that was fit into the parallel scheme. In addition, feature space sampling is used as well to reduce computational complexity, and to reach additional speedup. The system was implemented on a many-core GPGPU platform in order to observe the performance gain of the data-parallel construction. The chapter discusses the evaluation made on a public benchmark and the numerical results proving that the system performs well among other data-driven algorithms. Additionally, detailed assessment was done using real-life, high resolution images to confirm that the segmentation speed of the parallel algorithm improves as the number of utilized processors is increased, which indicates the scalability of the scheme.

DOI:10.15774/PPKE.ITK.2013.005

3.1 Introduction

Thanks to the mass production of fast memory devices, state of the art semiconductor manufacturing processes, and vast user demand, most present-day photo sensors built into mainstream consumer cameras or even smartphones are capable of recording images of up to a dozen megapixels or more. In terms of computer vision tasks such as segmentation, image size is in most cases highly related to the running time of the algorithm. To maintain the same speed on increasingly large images, the image processing algorithms have to run on increasingly powerful processing units. However, the traditional method of raising core frequency to gain more speed—and computational throughput—has recently become limited due to high thermal dissipation, and the fact that semiconductor manufacturers are attacking atomic barriers in transistor design. For this reason, future development trends of different types of processing elements—such as digital signal processors, field programmable gate arrays or general-purpose computing on graphics processing units (GPGPUs)—point towards the development of multi-core and many-core processors that can face the challenge of computational hunger by utilizing multiple processing units simultaneously [74].

The interest of this chapter is centered around the task of fast image segmentation in the range of quad-extended, and hyper-extended graphics arrays. The following sections describe the steps of design, implementation and numerical evaluation of the proposed segmentation framework that works in a data-parallel way, and can therefore efficiently utilize many-core mass processing environments. The structure of the system follows the bottom-up paradigm and can be divided into two main phases. During the first, clustering step, the image is decomposed into sub-clusters. Deriving the consequences from the analysis of data-driven algorithms (see Subsection 2.3.1), the core of this step is based on the mean shift segmentation algorithm that was embedded into the parallel environment, allowing it to run multiple kernels simultaneously. The second step is a cluster merging procedure that joins sub-clusters that are adequately similar in terms of color and neighborhood consistency.

At this point of research, my main aim was not to exceed the quality of the original mean shift procedure. Rather, to show that by a giving parallel extension of the mean shift algorithm good segmentation accuracy can be achieved with considerably lower running time than the serial implementation that operates with a single kernel at a time.

DOI:10.15774/PPKE.ITK.2013.005

To be able to evaluate its segmentation potential, the framework has been implemented on a GPGPU platform and numerical evaluation was run on miscellaneous GPGPUs with different numbers of stream processors to demonstrate algorithmic scaling of the clustering step and speedup in segmentation performance.

3.2 Computational Method

As it was discussed in 2.4, the weakness of the mean shift is that its running time is quadratically proportional to the number of image pixels. This property makes it slow, especially when working with large images. Three main acceleration strategies were used to speed up the procedure:

1. Reduce the computational complexity by sampling the feature space (see Subsection 3.2.1);
2. Gain speedup through the parallelizing the inner structure of the segmentation (see Subsection 3.2.4);
3. Reduce the number of mean shift iterations by decreasing the number of saturated kernels required for termination (referred to as abridging) (see Subsection 3.2.5).

Figure 3.1 reveals the sematic flowchart of the segmentation framework.

3.2.1 Sampling Scheme

The motivation behind sampling is straightforward: it reduces the computational demand, which is a cardinal aspect in the million-element feature space domain. The basic idea is that instead of using all n feature points, the segmentation is run on $m \ll n$ initial elements. The mean shift iteration is then started from these seed points, and the other elements of the feature space are assigned to the so-obtained modes by using certain local rules [37, 46, 53, 55, 58, 75].

There are however two major factors one has to take into account in the case of sampling: undersampling the feature space can highly decrease segmentation quality, while oversampling leads to computational and memory-related overheads.

To address the above concerns, a recursive sampling scheme was designed that works as follows.

DOI:10.15774/PPKE.ITK.2013.005

3. PARALLEL FRAMEWORK

1. Initialize a mean shift kernel in a yet unclustered element i of the feature space and repeat the mode seeking iteration until termination that is denoted by t_{sat}.

2. At this point, FSE χ_j is assigned to a $\psi_i = \chi_i^{t_{\text{sat}}} = (\mathbf{x}_{r,i}^{t_{\text{sat}}}; \mathbf{x}_{s,i}^{t_{\text{sat}}})$ mode that is obtained from χ_i sampled initial mean shift centroid if, and only if

$$\|\mathbf{x}_{s,j} - \mathbf{x}_{s,i}^{t}\| < h_s, \tag{3.1}$$

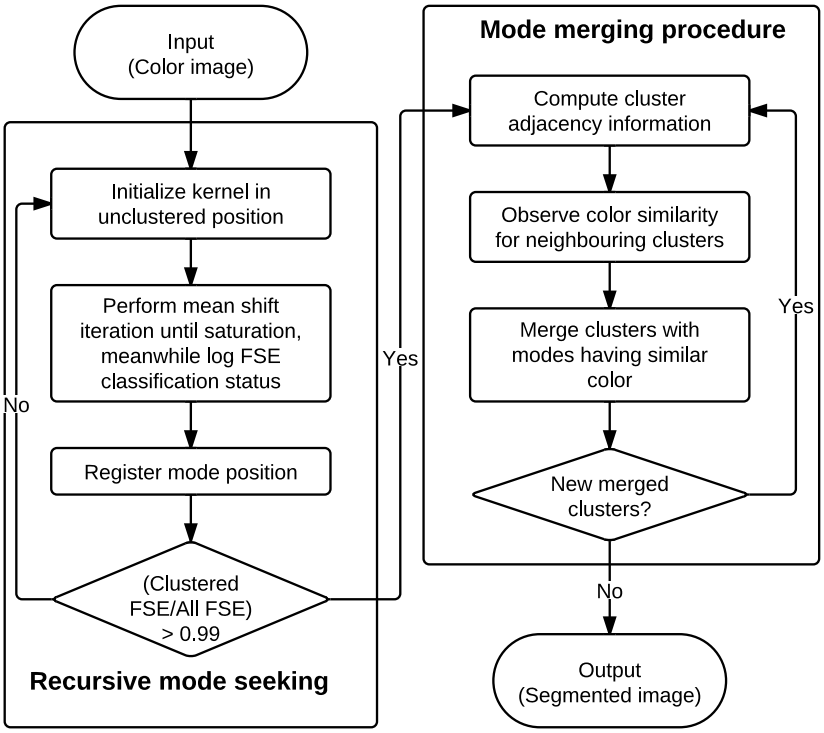


Figure 3.1: Flowchart of the segmentation framework. The result of the recursive mode seeking procedure is a clustered output that is an over-segmented version of the input image. The step of mode seeking is therefore succeeded by the merging step that concatenates similar clusters such that a merged output is obtained. The term FSE refers to feature space element.

DOI:10.15774/PPKE.ITK.2013.005

and

$$\|\mathbf{x}_{r,j} - \mathbf{x}_{r,i}^{t}\| < h_r, \tag{3.2}$$

where $t \in [0, t_{\text{sat}}]$. In case a pixel is covered by more than one kernel, it is associated to the one with the most similar color.

3. If unclustered FSEs remain after the pixel-cluster assignment, resampling is done in the joint feature space, and new mean shift kernels are initialized in those regions, in which most unclustered elements reside.

3.2.2 Dynamic Kernel Initialization

Since resampling is driven by the progress of the clustering of the feature space, both the *number* and the *position* of mean shift kernels is selected in proportion to the content of the image. Note that in the case of real-life images, the image usually contains high frequency shading and gleams due to inconsistent lighting conditions. These phenomena appear in the feature space as outliers. For this reason a solution similar to the smallest significant feature size proposed by Meer and Comaniciu (see Subsection 2.3.3) was applied. But instead of removing small classes from the fully clustered image in a post-processing step, resampling is terminated when the number of clustered elements in the feature space reaches 99%, at which point all unclustered elements are assigned to the closest mode.

3.2.3 Cluster Merging

After the iterative clustering procedure finished, cluster merging is performed. Two simple rules were used for concatenation: cluster i and j are joined if they satisfy the following two criteria:

C1. The two clusters have a common border in terms of eight-neighbor connectivity.

C2.

$$\|\mathbf{x}_{r,i} - \mathbf{x}_{r,j}\| < h_r, \tag{3.3}$$

where $\mathbf{x}_{r,i}$ and $\mathbf{x}_{r,j}$ are the range components of the modes of the corresponding clusters.

If both criteria hold for a pair of observed classes, the position of the mode in the feature space is recalculated as described by Equation 4.17 in Subsection 4.3.3.

DOI:10.15774/PPKE.ITK.2013.005

3.2.4 Parallel Extension

The recursive serial framework described in Subsection 3.2.2 was extended to work in a parallel way:

Step 1. Initialize a given number of mean shift kernels on the joint feature space.

Step 2.a Perform the iterative mode seeking procedure (Equation 2.3) of the concurrent kernels simultaneously until termination.

Step 2.b Perform pixel-cluster assignment according to Equation 3.1 and Equation 3.2 respectively and save the position of the obtained modes. In case an FSE is within the support of multiple kernels, assign it to the mode with the smaller distance (color, i.e. Equation 3.2 is prioritized).

Step 3. Observe the topology of unclustered elements:

- If the feature space requires additional clustering, go to step 1.
- If the feature does not require additional clustering, proceed to cluster merging.

Merging is performed after the clustering is finished, and it is also a recursive procedure:

Step 1. Compute pairwise neighboring information of the clusters (i.e. isolate clusters for which ***C1*** is true).

Step 2. Observe criterion ***C2*** for adjacent clusters:

- If ***C2*** does not hold for any cluster pair, terminate the merging procedure.
- Otherwise, continue with step 3.

Step 3.a Unify clusters for which both ***C1*** and ***C2*** hold by recalculating the feature space position of the class-defining mode using Equation 4.17.

Step 3.b Return to step 1.

DOI:10.15774/PPKE.ITK.2013.005

While the theoretical advantages of parallel systems are widely known, the parallel implementation of the mean shift algorithm results in a few drawbacks that do not occur in the serial version.

The most important aspect of the parallel implementation is the memory intensive behavior. The position of a given mean is calculated using Equation 2.3 on the elements residing within the support/region of interest (ROI) window[3] of the kernel. However, the feature space elements grouped by the different ROI windows are stored in non-consecutive places in the device memory. This pattern does not favor coalescent memory access directly, which slows down the simultaneous mode seeking procedure. To accelerate these ROI operations, the ROI windows of a given mode seeking step are "cut" from the feature space and stored in a continuous structure.

The implementation induces another important change in the mean shift scheme. When running the mode seeking process given by Equation 2.3 on multiple autonomous kernels at once, it is not feasible to isolate saturated modes and replace them with new kernels in a "hot swap" way, due to the characteristics of block processing. Although such a switching solution is theoretically possible, it involves a lot of additional memory operations, which have a negative influence on the speed of the segmentation procedure. For this reason, a new mean position is calculated for each of the kernels utilized by the current sampling operation, until Equation 2.5 is met by every single one of them. Since this property is not present in the sequential mean shift, two important remarks should be made here.

1. This property does not result in corruption concerning the retrieval of image content. Kernels for which the shift of the mean value is below the threshold (Equation 2.5) will continue stepping towards the steepest ascent [40].

2. This property results in an overhead in terms of computational complexity.

3.2.5 Abridging Method

In order to suppress the number of redundant iterations (in other words, the number of additional steps of the kernels that are beyond saturation), a so-called *abridging method* was introduced.

[3]Note: the Gaussian kernel was used (see Subsection 3.3.3). Since it comes with an infinite support, a ROI window was selected—see Subsection 4.2

DOI:10.15774/PPKE.ITK.2013.005

3. PARALLEL FRAMEWORK

The method uses a single constant called the *abridging parameter* $A \in [0, 1]$ that specifies the minimum proportion of kernels that is required to saturate. At the time instant this value is met, the ongoing mode seeking procedure is terminated and the next resampling iteration is initialized.

The usage of abridging is demonstrated through the following example (see Figure 3.2): consider a parallel mode seeking procedure that is performed on n' kernels simultaneously (where $m = \sum n'$, see Subsection 3.2.1), and let us say that it takes t_{ter} mean shift iterations until all kernels saturate. The ratio of kernel saturation follows an exponential pattern, such that a remarkable fraction of the kernels saturates in the first few shifting steps, so that in their case, each additional iteration is superfluous. The abridging parameter gives us a simple tool to terminate the mode seeking procedure after a reasonable amount of steps, when the number of saturated kernels is satisfactory.

The practical effect of the abridging parameter was studied by running the segmentation on the "test set" of the then available version of the Berkeley Segmentation Dataset and Benchmark (BSDS300) [62] consisting of 100 images. Various kernel bandwidths and abridging parameters were used and the running times, the number of mean shift-, and *resampling iterations*, and the quality of output among other characteristics (see Subsection 3.3.4 for a detailed list of the parameter settings used) were measured. The following list gives an overview of the analytical aspects and the main conclusions, whereas a more detailed description is given in the subsections where noted.

1. **Impact on the number of mean shift iterations.**
   The main motivation for using the abridging method is its strong effect of reducing the number of mean shift iterations. Compared to a setting of $A = 1$, a framework with $A = 0.6$ requires 3.1 times less mean shift iterations on average. Applying $A = 0.6$, this reduction was at least 2.04 times in 95% of the cases. The measured standard deviation of 0.79 underlines that the speedup is stable and present at a broad selection of bandwidths.

2. **Impact on the number of resampling iterations.**
   Abridging increases the number of resampling iterations, but has a small and strictly monotonically decreasing effect that is inversely proportional to the bandwidth parameters. The number of resampling iterations showed an increase of

DOI:10.15774/PPKE.ITK.2013.005

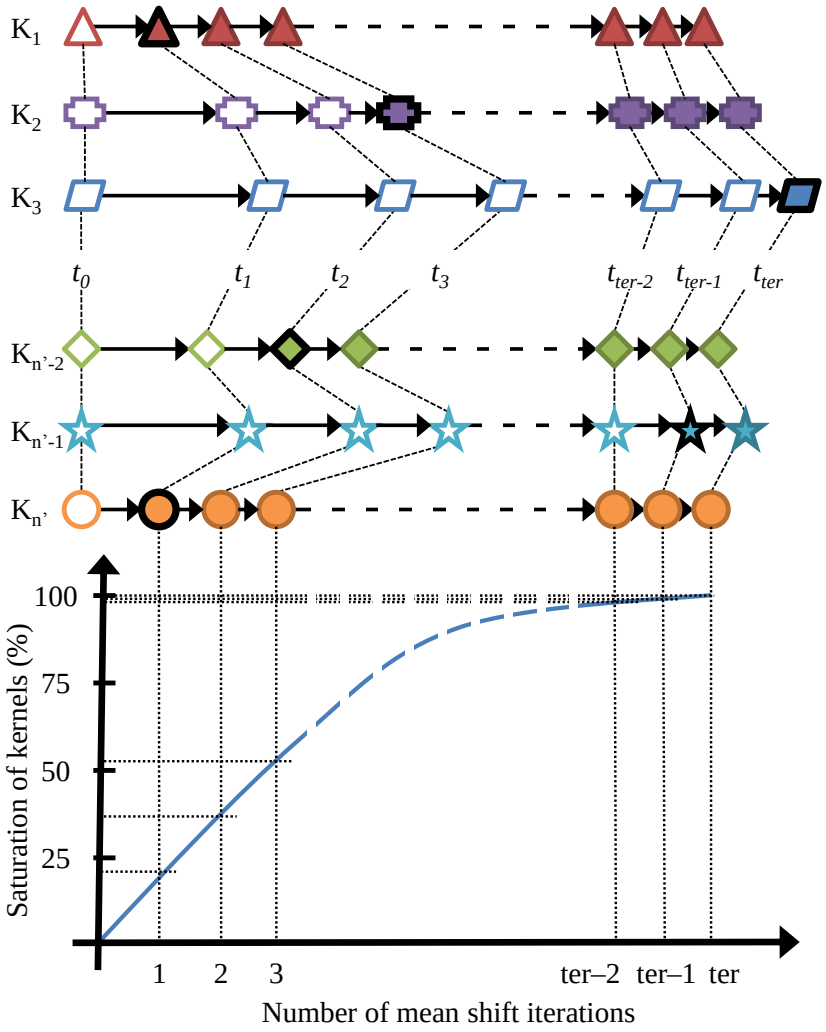


Figure 3.2: Demonstration of the tendency of kernel saturation. The upper half of the figure gives an example for a mode seeking procedure with n' simultaneously iterated kernel windows. Consecutive iterations are denoted by t, and the length of the arrows are proportional to the length of the shift of the centroid of the given kernel. t_{ter} is the iteration in which all kernels have saturated. Saturated kernels are filled, with a thick black silhouette highlighting the first such iteration. Each consecutive iteration for that kernel is redundant. The bottom half illustrates the number of mean shift steps vs. the percentage of saturated kernels.

DOI:10.15774/PPKE.ITK.2013.005

1-15% on average in a system with an abridging parameter of 0.6, which corresponds to 0.02-1.77 additional resamplings depending on the selected bandwidth parameters.

3. **Impact on segmentation speed.**
   The usage of the abridging parameter reduces the time demand of the mode seeking procedure, because although it may increase the number of resampling operations, it drastically cuts back the number of required mean shift iterations. Subsection 3.4.2 gives a complete overview.

4. **Impact on output quality.**
   The position of the mean values of kernels that did not saturate at the instant the abridging parameter caused termination are not situated at the local maxima of the underlying probability density map. Due to the proposed pixel-cluster assignment scheme, this only implies the formulation of clusters that have more localized color information, and in practice, it appears in the form of a slight over-segmentation. See Subsection 3.4.1 for a complete numerical evaluation.

5. **The actual number of saturated kernels.**
   The ratio of kernels saturated at termination generally exceeds the prescribed threshold ratio by 15-28% on average.

3.3 Experimental Design

One of the most important tasks within a data-parallel environment is the control of the simultaneous data access. In contrast to a simple threaded serial system, in which processing consists of consecutive—and thus: mutually exclusive—read and write memory accesses, a parallel environment requires additional buffering steps to properly handle simultaneous memory operations, and additional memory space to feed the processors.

Another issue with data-parallel programming is that compared to accesses to local memory on the device, the host to device memory transfers (and vice versa) are slow. For this reason, fitting the data representation into device memory is a key task in terms of speed.

DOI:10.15774/PPKE.ITK.2013.005

Lastly, limitations in the size of quickly accessible device memory calls for compact data representation, which again costs memory operations, and therefore time.

For the reasons listed above, parallelization of a given algorithm can only be considered effective if the speedup can be achieved in spite of all the enumerated constraints, and without sacrificing accuracy.

The proposed framework was analyzed concerning three different aspects:

1. the quality of the output (see Subsection 3.4.1),
2. the time demand of the algorithm on images with different size (see Subsection 3.4.2), and
3. the scaling on different devices with various number of processors (see Subsection 3.4.3).

Quality analysis was done with a broad selection of parameters in an exhaustive search-like scheme that has two notable benefits:

1. A broad overview about the robustness of the framework's output quality was obtained.
2. Optimal parametrizations both in terms of speed and quality were obtained that were used for the two alternative evaluation settings during the running time measurements.

3.3.1 Hardware Specifications

The parallel hardware architecture for the measurements was the GPGPU platform offered by NVIDIA. The measurements were performed on five GPGPUs with various characteristics. As a reference, the framework was also tested on a PC equipped with 4GB RAM and an Intel Core i7-920 processor clocked at 2.66GHz, running Debian Linux. The technical specifications of the hardware are summarized in Table 3.1. Note that in the case of the NVIDIA S1070, only a single GPU was utilized (for this reason it is referred later on as S1070SG).

Compute capability numbers consist of two values: a major revision number that is indicating fundamental changes in chip design and capabilities, and a minor revision number referring to incremental changes in the device core architecture.

DOI:10.15774/PPKE.ITK.2013.005

Table 3.1: Parameters of the used GPGPU devices.

Device name	No. of stream processors	Clock frequency	Device memory	Compute capability
8800GT	112	1500 MHz	1024 MB	1.1
GTX280	240	1296 MHz	1024 MB	1.3
S1070SG	240	1440 MHz	4096 MB	1.3
C2050	448	1500 MHz	3072 MB	2.0
GTX580	512	1544 MHz	1536 MB	2.0

3.3.2 Measurement Specifications

In the case of the scaling and timing experiments, the measurements were made on five different image sizes. The naming conventions and corresponding resolutions are summarized in Table 3.2.

Table 3.2: Naming convention and resolution data of the images used for the timing and scaling measurements.

Name of extended graphics array	Abbreviation	Resolution	Resolution in megapixels (MP)
Wide Quad	WQXGA	2560×1600	4.1 MP
Wide Quad Super	WQSXGA	3200×2048	6.6 MP
Wide Quad Ultra	WQUXGA	3840×2400	9.2 MP
Hexadecatuple	HXGA	4096×3072	12.6 MP
Wide Hexadecatuple	WHXGA	5120×3200	16.4 MP

3.3.3 Environmental Specifications

All measurements were performed in the 5D joint feature space consisting of the Y, Cb and Cr color coordinates, and (x, y) spatial position of each pixel. Color channels were normalized into the $[0, 1]$ interval, but the luminance channel was given an additional multiplier of 0.5 in order to somewhat suppress the influence of gradients that are often caused by the natural lighting conditions. The same normalization factor was used for both spatial channels, so that for a non-square image, the longer side is normalized to

DOI:10.15774/PPKE.ITK.2013.005

$[0, 1]$, whereas the maximum of the shorter side is the aspect ratio of the two sides. This way the isotropic property and the central symmetry of the kernel suggested by Meer and Comaniciu [1] was ensured.

The kernel window was selected to be the Gaussian, with distinct h_s and h_r parameters for the spatial and range domains respectively. To speed up the segmentation, the spatial weight kernel was calculated only once at the beginning of the segmentation, and was shifted to the position of the corresponding mode in each iteration. (Note: since the support of the Gaussian kernel is infinite, it is considered only within a radius in which its value is above 0.1—see Subsection 3.2.3.)

3.3.4 Quality Measurement Design

Since neither the BSDS500, nor the WIDB was published at the time when the quality measurements of the parallel system were done, the "test" set of the BSDS300 consisting of 100 pictures was used to provide quantitative results that are comparable with other algorithms. This set was segmented multiple times using the same parametrization for each image in a run. Three parameters were alternated among two consecutive runs: h_r taking values between 0.02 and 0.05, h_s with values in the interval of 0.02 and 0.05, both utilizing a 0.01 step size, and the abridging parameter ranging from 0.4 to 1.0 with a step size of 0.2. In each case, the segmenter was started with 100 initial kernels, and in every resampling iteration 100 additional kernels were utilized.

Note that since the BSDS300 benchmark evaluates quality based on boundary information, soft boundary maps were generated in the following way: the luminance channel of the output of the segmentation framework was subject to morphological dilation using a 3x3 cross-shaped structuring element. The difference of the original and the dilated channel resulted in an intensity boundary map.

The quality of the output was assessed using the F-measure values (see Equation 2.15).

3.3.5 Timing Measurement Design

Timing measurements aimed at registering the running time of the algorithm on high resolution real-life images. An image set consisting of 15 high quality images was formulated and the images were segmented in five different resolutions using the parameter settings "speed" and "quality" that were obtained during the quality measurements

DOI:10.15774/PPKE.ITK.2013.005

(see Subsection 3.4.1). In each case, the segmenter was started with 10 initial kernels, and 10 additional kernels were utilized in every resampling iteration.

3.3.6 Scaling Measurement Design

The mean shift iteration specified in Equation 2.3 was timed individually on the different devices (and as a reference, on the CPU) to observe the scaling of the data-parallel scheme. To give a complete overview, all linear combinations of spatial bandwidth parameters ranging from 0.02 to 0.05 with a step size of 0.01, and kernel numbers of 1, 10 and 20 were measured. Each value in the corresponding figure represents a result that was obtained by averaging 100 measurements (see Figure 3.8).

3.4 Results

3.4.1 Quality Results

As a result of alternating h_r, h_s and the abridging parameter, the framework was run with 64 different parametric configurations for each image of the 100 image BSDS300 test corpus.

The obtained average F-measure values for the different bandwidths and abridging parameters are displayed in Figure 3.3.

The highest F-measure value was 0.5816 for parameters $h_r = 0.03$ and $h_s = 0.02$ without any abridging, which fits in well among purely data-driven solutions [24]. It can be observed on Figure 3.3 that the output quality remained fairly consistent when relatively small bandwidths were selected. The system is more robust to changes made to the spatial bandwidth, while selecting a high range bandwidth parameter decreases output quality. As one may expect, abridging has a negative effect on quality, but it can be seen that for certain parameter selections (namely, for $h_s \in [0.02, 0.03]$ and $h_r \in [0.03, 0.04]$) even an abridge level of 0.6 results in acceptable quality. An interesting observation is that when both bandwidth parameters are set high, smaller abridging parameter values increase quality. The explanation for this is the following: as described in Subsection 3.2.5, abridging induces over-segmentation, and in this context, has an effect similar to having a smaller bandwidth parameter. This way, additional edges appear in the soft boundary map, from which many are coincident with the ground truth references used by the benchmark.

DOI:10.15774/PPKE.ITK.2013.005

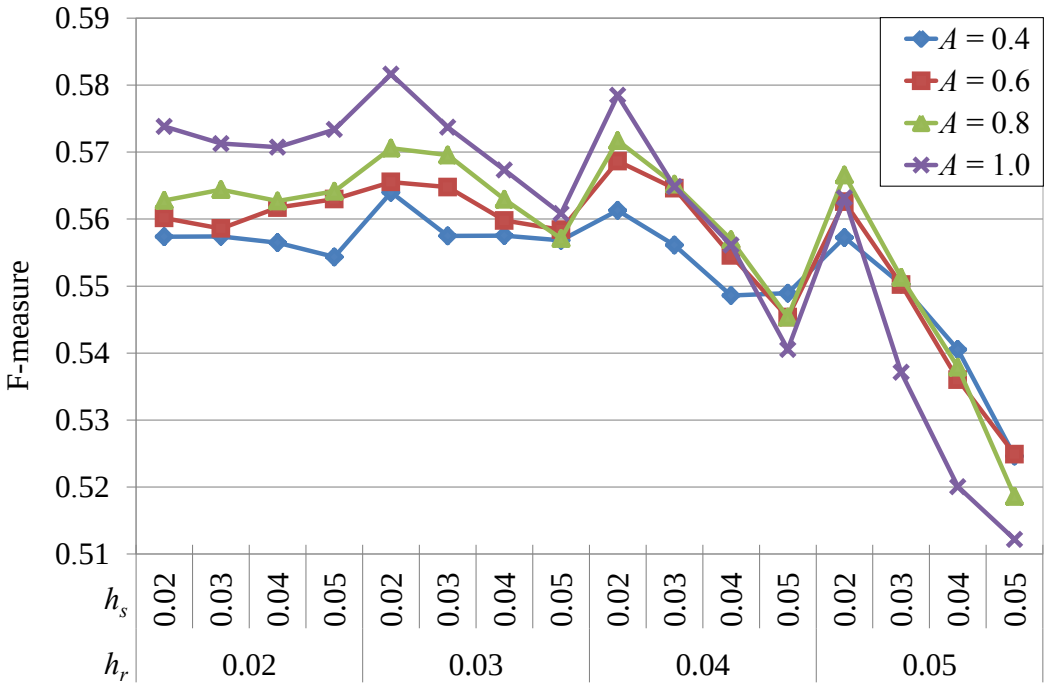


Figure 3.3: F-measure values obtained for the different parametrizations of the segmentation framework. h_s and h_r denote the spatial and range kernel bandwidths respectively, A values stand for different abridging constants.

Table 3.3 displays the F-measure values for the different parametric constellations given as the percentage of the best result.

The presence of a gray background indicates that quality loss is less than 3% compared to the best result. Based on these results, two parametric settings were selected for the timing measurements:

- the **Quality setting** was selected to be $(h_s, h_r, A) = (0.02, 0.03, 1)$, while
- the **Speed setting** was selected to be $(h_s, h_r, A) = (0.03, 0.04, 0.6)$.

In the case of the quality setting the only guideline was to obtain the best quality, whereas in the case of the speed setting the preferences in order of precedence were the quality (should be better than 97%), the value of the abridging constant (a smaller parameter makes segmentation faster), finally the size of the bandwidth parameters (bigger is faster due to the data-parallel structure and the formulation of the pixel-cluster assignment scheme—see Equations 3.1 and 3.2).

DOI:10.15774/PPKE.ITK.2013.005

Table 3.3: F-measure values obtained with different abridging and bandwidth parametrization given as the percentage of the best result. The presence of a gray background indicates that quality loss is less than 3% compared to the best result. The two settings selected for performance evaluation are denoted by bold letters. Measurements were made on the test set of the BSDS300.

h_r	h_s	$A = 0.4$	$A = 0.6$	$A = 0.8$	$A = 1.0$
0.02	0.02	95.83%	96.31%	96.76%	98.66%
	0.03	95.84%	96.04%	97.04%	98.22%
	0.04	95.67%	96.58%	96.75%	98.12%
	0.05	95.31%	96.80%	97.00%	98.58%
0.03	0.02	96.97%	97.23%	98.10%	**100%**
	0.03	95.85%	97.10%	97.92%	98.64%
	0.04	95.86%	96.24%	96.79%	97.54%
	0.05	95.73%	96.01%	95.78%	96.42%
0.04	0.02	96.50%	97.77%	98.30%	99.46%
	0.03	95.62%	**97.07%**	97.18%	97.12%
	0.04	94.32%	95.35%	95.75%	95.61%
	0.05	94.38%	93.76%	93.76%	92.94%
0.05	0.02	95.81%	96.73%	97.41%	96.83%
	0.03	94.61%	94.60%	94.79%	92.35%
	0.04	92.93%	92.16%	92.48%	89.41%
	0.05	90.20%	90.24%	89.16%	88.06%

Figure 3.4 shows a few example images from the 100 image BSDS300 "test" segmentation corpus among with the segmented output and the obtained F-measure for both the quality and the speed settings.

3.4.2 Running Time Results

The average running times measured on the 15 image corpus are summarized in Figure 3.5 using the quality and speed settings.

In the case of the measurements made on the GPGPUs, the displayed values include all operations and memory accesses that have been performed in order to obtain the merged output image. The clock was started before the host to device data transfer

DOI:10.15774/PPKE.ITK.2013.005

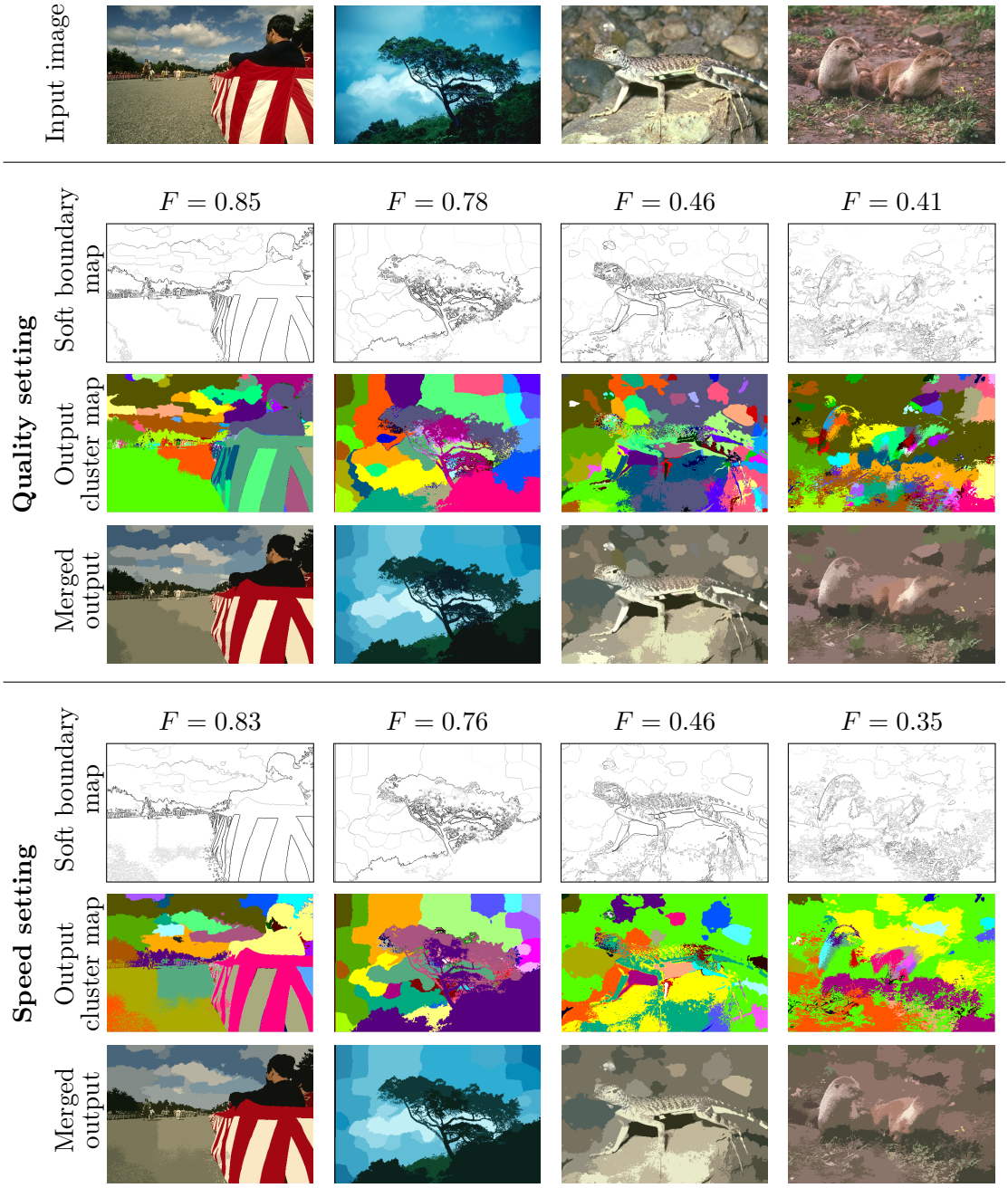
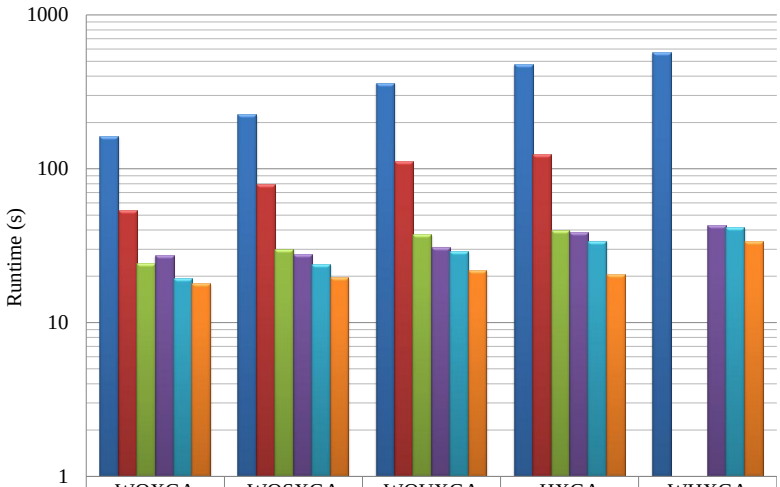


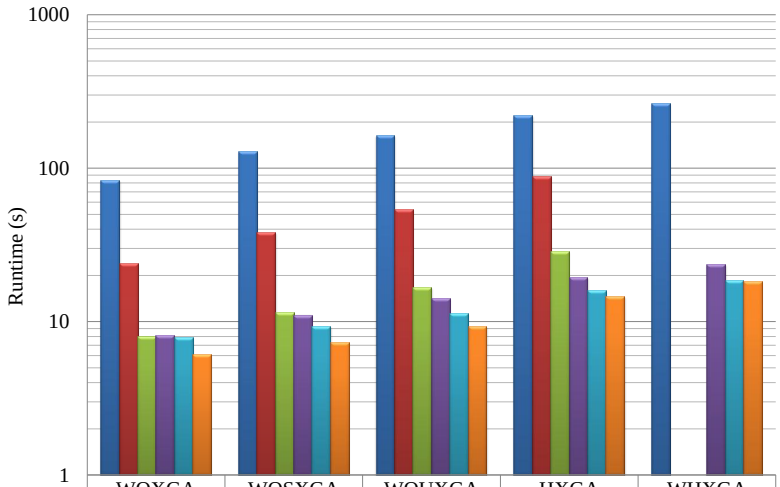
Figure 3.4: Four segmentation examples from the 100 "test" image corpus of the BSDS300. Results for both the quality and the speed setting are shown. Quality evaluation was run using the soft boundary map generated from the merged output. For the sake of better visibility of the extent of the clusters, they are displayed in the form of cluster maps as well. The obtained F-measure values are denoted by F.

DOI:10.15774/PPKE.ITK.2013.005

3. PARALLEL FRAMEWORK



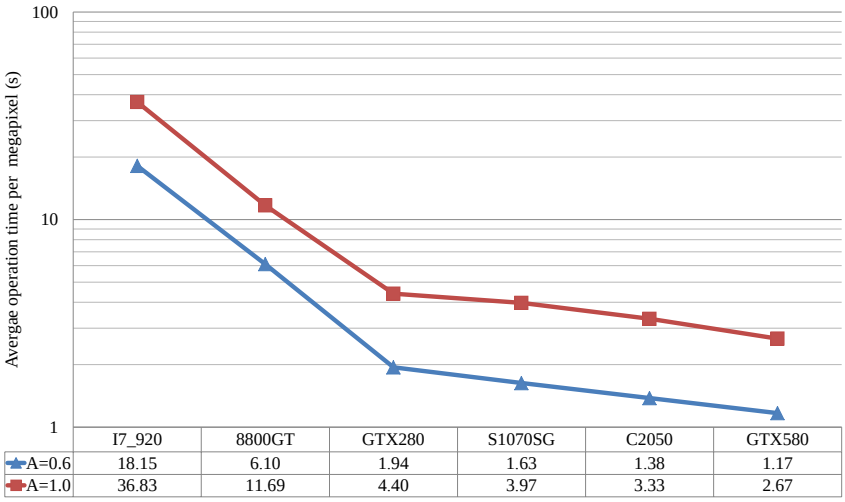
	WQXGA	WQSXGA	WQUXGA	HXGA	WHXGA
I7_920	160.46	224.63	359.60	471.86	565.09
8800GT	53.56	78.67	110.92	122.69	N/A
GTX280	23.95	29.96	37.26	39.80	N/A
S1070SG	27.27	27.75	30.84	38.16	42.80
C2050	19.32	23.62	28.85	33.73	41.38
GTX580	17.89	19.57	21.76	20.51	33.37



	WQXGA	WQSXGA	WQUXGA	HXGA	WHXGA
I7_920	82.95	127.92	163.05	220.17	261.52
8800GT	23.76	38.02	53.65	88.37	N/A
GTX280	7.99	11.48	16.55	28.64	N/A
S1070SG	8.14	10.97	14.16	19.34	23.39
C2050	7.79	9.24	11.25	15.78	18.46
GTX580	6.04	7.30	9.23	14.45	18.23

Figure 3.5: Running time values of the algorithm run on images with different sizes using five different GPGPUs and the CPU as the reference. Displayed are running time results obtained using the quality setting (top) and the speed setting (bottom). Each measurement displays an average value obtained from running the algorithm on 15 images. In case when "N/A" values are displayed, the onboard device memory sometimes became a bottleneck, which resulted in frequent caching operations that seriously slowed down the segmentation performance.

DOI:10.15774/PPKE.ITK.2013.005



	I7_920	8800GT	GTX280	S1070SG	C2050	GTX580
A=0.6	18.15	6.10	1.94	1.63	1.38	1.17
A=1.0	36.83	11.69	4.40	3.97	3.33	2.67

Figure 3.6: Average running time of clustering one megapixel on the different devices (and on the CPU) as a function of the abridging parameter. Difference in quality between the two settings is 3%.

carrying the input image started, and was stopped after the device to host data transfer carrying the merged output was completed, such that the output was retrieved into host memory. The same rule applies to the CPU measurements, but in this case obviously neither host to device transfers nor device to host transfers were necessary.

When using either the GTX580 or the C2050, the average time demand for segmenting a 16 megapixel image was just above 18 seconds in the case of the speed setting, and a bit more than 33 seconds on the GTX580 using the quality setting. Compared to the running times of the CPU using the same 16 megapixel setup (as utilized on the GTX580), this means an acceleration of 16.93 times in the case of the quality setting, and an acceleration of 14.34 times in the case of the speed setting. Figure 3.6 displays the time spent on average to cluster a million pixels on the different platforms. It can be seen that by sacrificing 3% of the quality, double speed can be achieved in most of the cases.

Figure 3.7 shows an example of the high quality input images from the 15 image segmentation corpus and the segmented output before and after the merging procedure. Two additional examples can be found in Appendix A.

DOI:10.15774/PPKE.ITK.2013.005

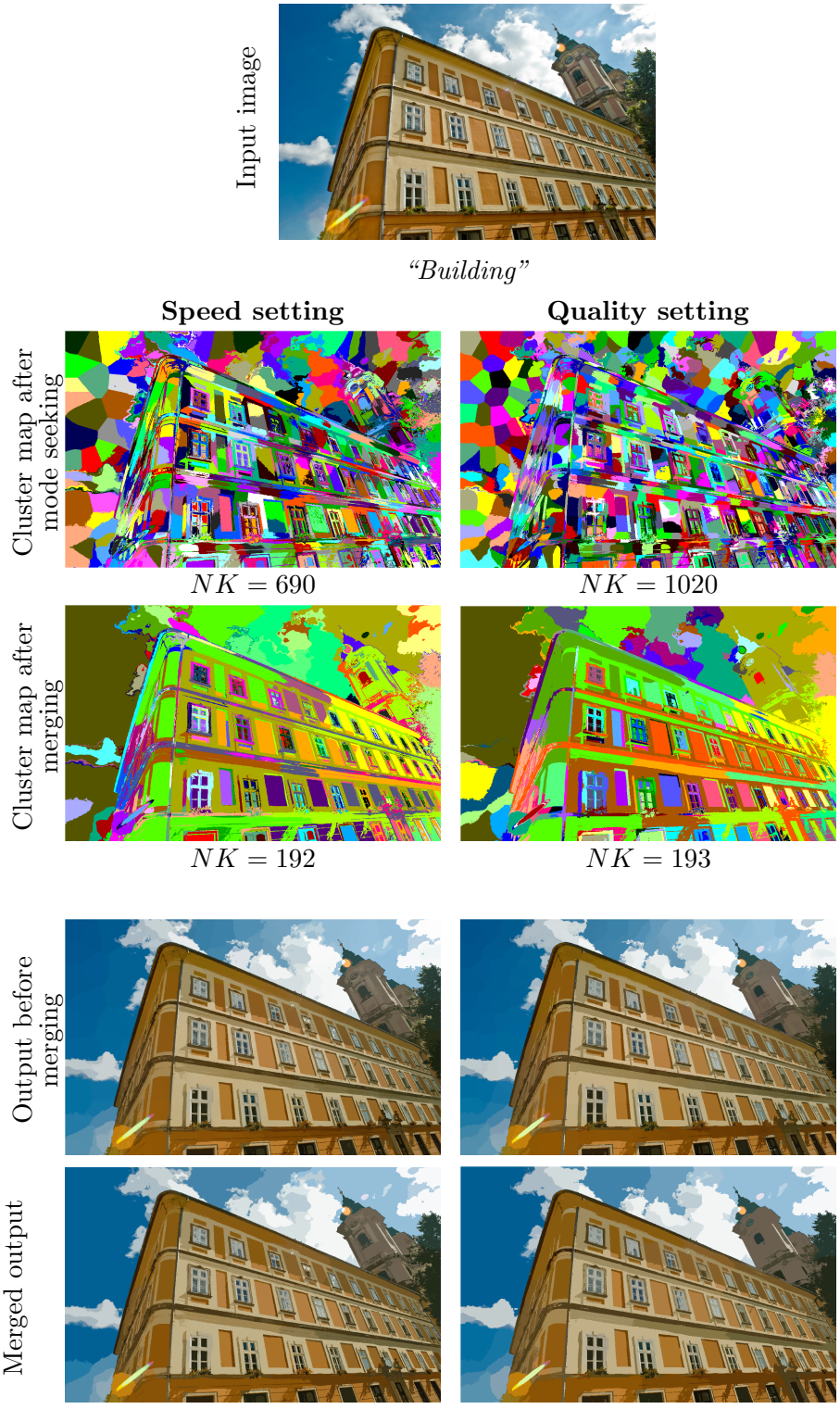


Figure 3.7: A high resolution segmentation example from the 15 image corpus used for the evaluation of the parallel framework. For the sake of better visibility, the extent of the clusters is also displayed in the form of cluster maps before and after the merging procedure. NK refers to the number of clusters.

DOI:10.15774/PPKE.ITK.2013.005

3.4.3 Scaling Results

As a result of the different parametrizations, the mean shift iteration was timed using 60 different constellations on the 5 GPGPU devices (and additionally, the CPU) with each measurement indicating an average value recorded on 100 iterations (as described in Subsection 3.3.6).

The first aspect of evaluation was the scaling of performance on the different GPGPU device generations. Figure 3.8 displays the obtained running time in milliseconds for a single kernel measured using different resolutions with bandwidth parameters $h_s = 0.02$ and $h_s = 0.05$. The running times show a clear tendency: as a result of improved characteristics (such as the number of stream processors, memory handling, caching and in some cases, operating frequency), the performance of newer device generations is superlinear compared to the older ones.

The second aspect of evaluation was the robustness of the operating time demand related to the number of kernels. To obtain expectations for a linear running time demand, the running time results measured when utilizing a single kernel were multiplied with 10 and 20 respectively. These expected values were then subtracted from the measured running time results and the outcome was evaluated for each device and the CPU. Table 3.4 displays the obtained results. On this table it can be seen that in the case when using 20 kernels, the maximum difference is negative for all GPGPUs. This means that the measured running time performance is always better than the expected one. In this context however there are exceptions, when 10 kernels were used. However in this cas the average difference is negative for all of the devices, which indicates that on average, the running time benefit is present. The closer this value to is zero, the more robust the running time on the used device with respect to the alternation of the number of kernels is.

Finally, the running time of calculating the mean shift iteration on the different devices was investigated in proportion to the running time of the same task measured on the CPU. Figure 3.9 displays an overview of the speedup that is obtained by taking into account all of the different parametrizations of $h_s \in [0.02, 0.05]$ and the number of kernels being 1, 10 and 20.

As one may expect, the fastest performance was observed on the GTX580: compared to the CPU, the speed increase was greater than 28 for all parameter settings, with

DOI:10.15774/PPKE.ITK.2013.005

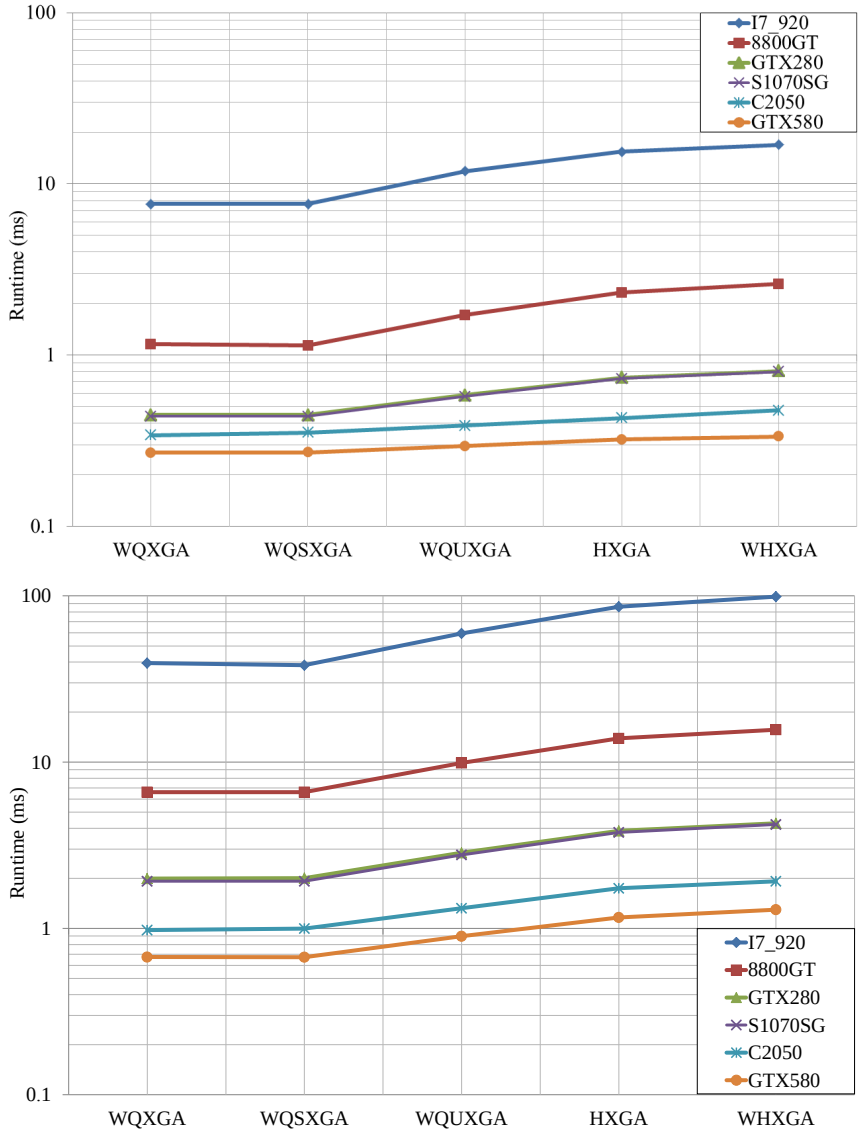


Figure 3.8: Running time tendencies of one mean shift iteration of a single kernel measured on the different devices (and the CPU) using different resolutions. Spatial bandwidth selection was 0.02 in case of the top figure and 0.05 in case of the bottom one. Bandwidths within this spatial domain follow the same running time pattern.

an average speedup of around 120. One may ask why the speedup of the mean shift iteration differs from the overall speedup of the framework. The answer to this question is that in the case of the former, only arithmetic operations are involved, thus these results represent the speed of the GPGPU processing units more closely. In contrast,

DOI:10.15774/PPKE.ITK.2013.005

Table 3.4: The robustness of the scaling on the different devices and the CPU. The statistics display the values obtained by comparing the expected running time values (derived by a multiplying the running time measured when using a single kernel) and the corresponding measured values. The relative standard deviation is the quotient of the standard deviation and the average.

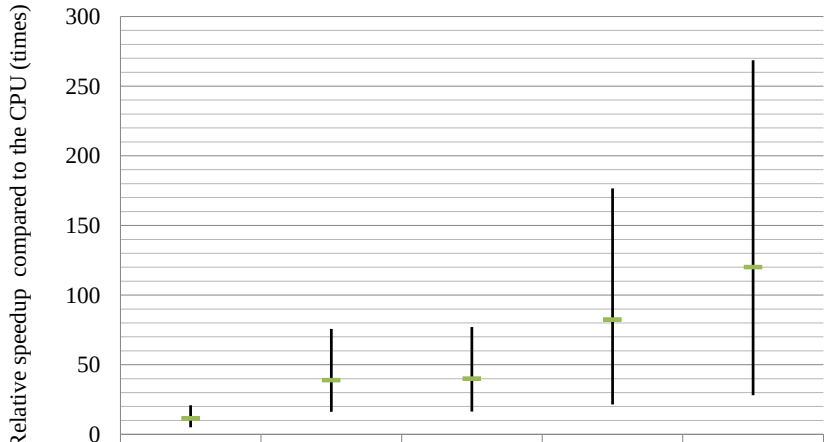
No. of used kernels	Device type	Relative standard deviation (%)	Minimum difference (ms)	Maximum difference (ms)	Average difference (ms)
10	I7_920	90.57	3.908	476.133	162.733
	8800GT	162.39	−12.461	2.402	−2.382
	GTX280	70.80	−0.773	1.118	−0.588
	S1070SG	60.86	−0.735	0.848	−0.567
	C2050	19.24	−1.228	−0.688	−0.815
	GTX580	17.05	−0.892	−0.480	−0.704
20	I7_920	77.17	140.130	3,436.867	1,210.004
	8800GT	95.69	−62.323	−4.469	−18.438
	GTX280	70.48	−14.218	−1.377	−5.745
	S1070SG	65.22	−14.070	−2.273	−5.877
	C2050	45.28	−7.765	−2.463	−4.282
	GTX580	34.21	−5.743	−2.199	−3.357

the overall speedup—with all the data transfers, memory read and write operations that are involved—represent the integrated performance of the device.

Three factors affect the observed speedup of the mean shift iteration, these are: the size of the image, the kernel bandwidth and finally the number of kernels. In order to clarify their individual effect, Figure 3.10 displays the influence of varying these parameters on the observed speedup.

Figure 3.10 shows a clear trend: the parameter with the most influence on raising the speedup is the number of kernels. This is resulted by the data-parallel nature of the task.

DOI:10.15774/PPKE.ITK.2013.005



	8800GT	GTX280	S1070SG	C2050	GTX580
Maximum speedup	20.90	75.71	77.14	176.61	268.59
Minimim speedup	5.09	16.24	16.46	21.54	28.17
Average speedup	11.35	38.88	40.00	82.34	120.06

Figure 3.9: Speedup results obtained for different devices by pairwise comparison to the CPU. The basis of comparison were the running time values representing the time demand of calculating new position(s) of mean(s) with all combinations of $h_s \in [0.02, 0.05]$ with number of kernels being 1, 10 and 20.

3.5 Conclusion

This chapter discussed the details and parallel design of the proposed segmentation framework. The core of this system is given by the parallel extension of the mean shift algorithm, that is accelerated by utilizing an abridging technique that can also be used in existing parallel mean shift techniques, such as [55, 56, 75], and a recursive sampling scheme that can narrow the complexity of the feature space, and is applicable in other solutions [59, 75] as well. The framework was implemented on a many-core computation platform, and a common segmentation benchmark was used to evaluate the output quality, and to demonstrate its robustness concerning parameter selection. Segmentation performance was analyzed on different high resolution real-life images, using five GPGPUs with miscellaneous specifications. The running time of a parallel mean shift iteration was measured on the different devices in order to observe the scaling of the data-parallel scheme. The algorithm has proven to work fast and to provide good

DOI:10.15774/PPKE.ITK.2013.005

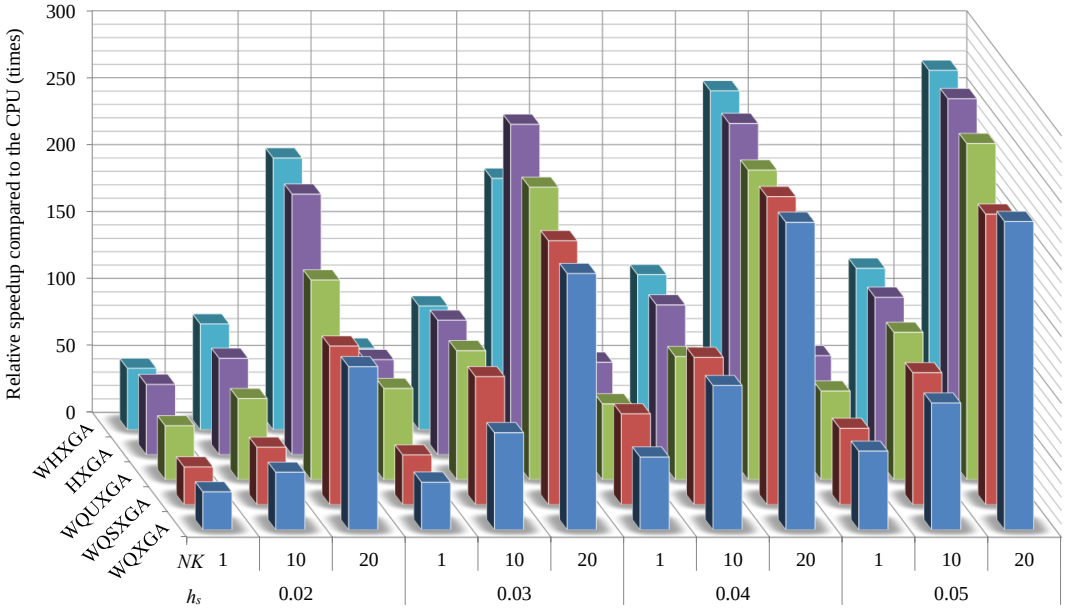


Figure 3.10: Speedup results of the GTX580 as a pairwise comparison to the CPU using different parameter settings. The bases of comparison were the running time values representing the time demands of calculating new mean positions. All combinations of $h_s \in [0.02, 0.05]$ with number of kernels (NK) being 1, 10 and 20 were tested.

quality outputs.

DOI:10.15774/PPKE.ITK.2013.005

3. PARALLEL FRAMEWORK

DOI:10.15774/PPKE.ITK.2013.005

Chapter 4

Adaptive Extension

This chapter discusses the method of how the building blocks of the parallel algorithm were extended to operate with respect to the content of the input image. In case of the segmentation phase, the bond confidence concept is introduced, which incorporates an intelligent sampling scheme and a nonlinear pixel-cluster assignment method. The proposed sampling can adaptively determine the amount and spatial position of the samples based on the local properties of the image and the progress of the segmentation. Sampling is driven by a single bond confidence value that is calculated without overhead during the mean shift iterations. The same parameter guides the pixel-cluster mapping that can ensure that each picture element is associated with a class having the most similar characteristics. The method of determining similarity in the merging phase has been extended to tolerate the rapid changes in intensity, hue, and saturation, which occur frequently in real-life images. The focus during the evaluation of the framework has been put onto output accuracy that is measured on three publicly available datasets using numerous metrics and a high resolution image set. The detailed results underline that the output quality of the framework is comparable to the reference but works an order of magnitude faster.

DOI:10.15774/PPKE.ITK.2013.005

4.1 Introduction

The motivation for the adaptive extension of the parallel segmentation framework arose during its evaluation made on the high resolution images.

It is straightforward that the resolution of an image directly influences the running time and the output accuracy of a segmentation algorithm. But in case of lossy algorithms, the change in these two characteristics is not totally explained by the resolution because the distribution of information in real-life images is very heterogeneous (see Figure 1.1). This property is a lot more emphatic on larger images than any other characteristic, not to mention that most segmentation datasets still prefer under-segmentation, and the suppression of details (see Subsection 2.5.3). On one hand some regions present in these images contain a lot more details than in those images of small resolution, but even more importantly, often there are many large surfaces that belong to the same object (or the background) and have the same, homogeneous representation in the feature space. Since the parallel algorithm explained in Chapter 3 uses feature sampling, it was straightforward that both computational efficiency and output quality could be improved, if the sampling scheme would be guided by heuristics that are built upon this property. However, it was kept in mind that in most cases, the calculation of an efficient heuristic costs additional arithmetic computation that can slow down the system, therefore it was cardinal to find a way to minimize the number of extra calculations required.

As it was discussed in Subsection 2.5.1, if we use higher resolution images not just for their own sake, the amount of details present in the image grows. In e.g. a classification task, the appearance of additional details make the description of objects more robust, but on the other hand, somewhat more complex as well. In case an object is composed of parts that have completely different characteristics and feature representations (to give the easiest example: the black and yellow butterfly in Figure 2.3, but one can also consider the man in the hawaiian shirt in Figure 4.6), proper segmentation could make use of high-level knowledge.

To handle the problems enumerated above, this chapter presents the following contributions:

1. **Multipurpose applicability.**

   The proposed framework returns a two-level output: the result of the data-driven

DOI:10.15774/PPKE.ITK.2013.005

segmentation that has a structure based purely on the characteristics of the image and the result of a subsequent merging process that utilizes a set of similarity rules. This scheme offers the possibility to directly inject alternative information (such as semantic, top-down metadata) or additional, task-dependent rules into the merging procedure with respect to the characteristics of the given task. Four algorithmic stages have been identified as potential entry points for knowledge-driven and/or task-dependent information.

2. **Reduced computational demand along with compact representation.**
   The segmentation algorithm utilizes adaptive sampling such that sampling frequency is based on the local properties of the image. Homogeneous image regions get clustered fast, initializing only a few large kernels, while spatially non-uniform regions, containing fine details are processed using more kernels of smaller sizes that provide extensive information. While preserving the content of the image, this intelligent scheme reduces both the computational requirement and the memory demand, enabling the segmentation of large images as well.

3. **High segmentation quality utilizing a nonlinear pixel-cluster mapping system.**
   Accuracy is pursued using a single-parameter system that registers the strength of the bond between a pixel and the mode of a cluster, subject to their spatial distance and color similarity. This way each picture element is associated with a class having the most similar characteristics. The key element for both the sampling procedure and the voting algorithm is the *bond confidence value*, which is calculated implicitly during the segmentation phase with no overhead.

4. **Fast operation due to parallel design.**
   All algorithmic extensions discussed in this chapter have been fit into the parallel framework, thus it still exploits the benefits of many-core platforms that can make the segmentation much faster, especially when dealing with a large amount of data.

Due to the observations discussed above, segmentation quality assessment became a major priority besides the segmentation speed measurements. Hence, it was a mandatory minimum for me to measure the capabilities of the enhanced system using various

DOI:10.15774/PPKE.ITK.2013.005

metrics offered by the public segmentation databases published subsequent to the assessment of the parallel framework. Additionally, this chapter evaluates the proposed method on a high resolution image set composed of over 100 images. But while in Chapter 3 the aim of these kind of measurements was to demonstrate algorithmic scaling, here the main dimension of evaluation is not how the alteration of resolution influences the running time, but how the varying amount of content does. It is shown via numerical analysis that the proposed adaptive framework can segment images with large homogeneous regions faster than a publicly available, non-adaptive variant, but at the same time the proposed system preserves many more details of complex image regions.

The following two sections explain the details of how the segmentation and the merging phases discussed in the previous chapter were enhanced to provide a better output quality.

4.2 Segmentation Phase

The segmentation phase is based on two major cornerstones: the *main equation of the mean shift method* that was already discussed in the previous subsection (Equation 2.3), and the *bond confidence concept* explained in this chapter.

The main idea behind this concept exploits that the utilized kernel function is essentially a low-pass filter that nonlinearly assigns a weight to all FSEs in an observed neighborhood around its center [76], such that the closer the FSE is to the mean subject to the considered feature space, the higher weight the kernel assigns to it. The bond confidence concept does not have restrictions for the mean shift theory regarding the type of the used kernel. A straightforward choice is the Epanechnikov kernel function due to its finite convergence that implicitly sets the ROI, i.e. the set of FSEs that are actually taken into account for the calculation of the new mean of the corresponding kernel. However, the Gaussian kernel was chosen due to its superior smoothing capabilities [77]. On the other hand it does not have a finite support, thus for the sake of efficiency, the ROI is defined to be a smaller window than the image itself, the radius of which is determined from Equation 2.4 in the following way:

$$r_{\mathrm{ROI}} = \sqrt{-2{h_s}^2 ln(\lambda)} \tag{4.1}$$

DOI:10.15774/PPKE.ITK.2013.005

where $0 < \lambda < 1$ is a user parameter, such that inside the ROI, the spatial kernel window generates values of $[\lambda, 1]$ depending on the distance from the mean of the window, and returns 0 outside the ROI[4]. As it will be discussed in this subsection, the dot product of the weighted color affinity and the weighted spatial locality is used as a pairwise *similarity metric* to determine the level of resemblance between the mean of the kernel and all the pixels within its support. If this value is sufficiently high in a mean shift iteration, the pixel is assigned to the kernel (referred to as *binding* in this dissertation), thus in the iteration when the mode of the kernel is found, its basin of attraction is directly given by the bound pixels.

Since numerous FSEs are "covered" by the support of a kernel, and the kernels are repositioned as the mean shift iterations are evaluated, in the classic mean shift method the majority of the kernels generally converges to trajectories already visited (as also noted in [54]), thereby performing calculations repeatedly. Again, if we initialize only $m \ll n$ mean shift kernels from adequate positions of the feature space, then a significant part of the redundant work can be saved (see Subsection 3.2.1). At this point there are two important questions to answer concerning this scheme:

1. How to select the *number* of the kernels (value of m)?

2. How to select the *initial positions* of the kernels in the feature space?

From the aspect of computational complexity, the obvious priority here is to minimize the number of samples. Simultaneously, we have to keep in mind that undersampling introduces loss of image details, whereas unnecessary over-sampling leads to computational and memory-related overheads along with a number of superfluous clusters that also raise the computational demand of the merging phase. Selecting the number of utilized kernels depending only on the resolution of the input image or alternatively, on the bandwidth parameter of the mean shift kernels will lead to a suboptimal segmentation result (additional discussion of this topic was given in Subsection 2.5.1).

If a set of neighboring pixels is homogeneous in color, then only a few kernels are required in the feature space to bind the FSEs that represent the pixels of this area. Thus, such regions are quickly processed with relatively low computational demand.

[4]Note: according to my experiments, the quality of the output of the framework is not sensitive to the choice of the λ parameter in a wide range. Experiments were performed in the $0.01 \leq \lambda \leq 0.2$ interval, and used $\lambda = 0.1$ for all measurements shown.

DOI:10.15774/PPKE.ITK.2013.005

On the contrary, an inhomogeneous region with many different colors and shades will require more kernels to be adequately segmented. These regions usually carry more localized information [64], therefore utilizing an increased number of kernels is also favorable, because it allows this information to be preserved in the segmented output.

However, prediction of the *optimal* number of kernels along with their initial position in the feature space would be way more complex than the segmentation procedure itself.

For this reason an iterative scheme was designed that determines both the number of kernels and their initial spatial positions dynamically, which can be considered a fair estimate.

The segmentation phase consists of L consecutive *loops* that are composed of the following triplet of steps in the displayed sequence:

Step 1. Adaptive sampling (see Subsection 4.2.1);

Step 2. Mean shift iterations and pixel binding (see Subsection 4.2.2);

Step 3. Pixel-cluster assignment check (see Subsection 4.2.3).

Before going into the details of the loop steps, it is important to observe the key element of the confidence concept: the *bond confidence*, denoted by $\Upsilon_j(\chi_i)$, represents the strength of the bond between FSE χ_i and kernel j. The bond confidence is determined as follows. Let us denote the set of all FSEs that reside within the support of kernel j with mean value χ_j^t:

$$S_j^t = \{\chi_i \in \mathcal{F} : i \in P_I, \|\mathbf{x}_{s,i} - \mathbf{x}_{s,j}^t\| \leq r_{\text{ROI}}\}. \tag{4.2}$$

The bond confidence $\Upsilon_j^t(\chi_i)$ in a given iteration $0 < t \leq t_{\text{sat}}$ for all $\chi_i \in S_j^t$, is given by

$$\Upsilon_j^t(\chi_i) = g\left(\left\|\frac{\mathbf{x}_{r,i} - \mathbf{x}_{r,j}^t}{h_r}\right\|^2\right) g\left(\left\|\frac{\mathbf{x}_{s,i} - \mathbf{x}_{s,j}^t}{h_s}\right\|^2\right). \tag{4.3}$$

After the saturation of the kernel, the overall bond confidence value between FSE i and kernel j can be determined:

$$\Upsilon_j(\chi_i) = \max_t(\Upsilon_j^t(\chi_i)). \tag{4.4}$$

One might easily recognize that Equation 4.3 is very similar to the denominator of Equation 2.3, and despite the lack of the final summation, it is being calculated

DOI:10.15774/PPKE.ITK.2013.005

during the standard mean shift iterations without introducing overhead. Due to the normalization term of the Gaussian, the range of the bond confidence is $[0, 1]$. A higher confidence value within this interval means a greater similarity to—and therefore a stronger association with—the mode of a kernel, such that a 0 indicates no affiliation, and 1 refers to a full match.

Also, prior to the start of the first loop, three data containers are initialized:

1. the *global mode vector* (GMV): incrementally stores the modes $\psi_j, \forall j \in [1, m]$ that are obtained throughout the loops;

2. the *global bond confidence* (GBC) matrix with $GBC_i = 0, \forall i \in P_I$: registers the highest bond confidence recorded during the segmentation phase

3. the *pixel-cluster mapping* (PCM) matrix with $\mathfrak{C}_i = \emptyset, \forall i \in P_I$: for each FSE it points to the cluster-defining kernel that generated the highest confidence value for it, which at the same time determines its final cluster assignment.

At the termination of each loop, the modes of new kernels are added to the GMV, furthermore the GBC and the PCM matrices are refreshed based on the corresponding in-loop values (defined in detail in steps 2 (Subsection 4.2.2) and 3 (Subsection 4.2.3)). Using the PCMs of the FSEs and the corresponding modes from the GMV the final color assignments can be determined after the segmentation phase is finished. As it will be discussed in the following, the bond confidence not only works as a similarity metric, but also as a discriminator during the steps of the adaptive sampling and the pixel-cluster assignment, thus unlike the case of most other algorithms that utilize sampling, no additional heuristics are required for these steps in the proposed system.

4.2.1 Step 1—Adaptive Sampling

Loop l begins with the initialization of m_l kernels, the set of which is denoted by K_l. The initial seed point of all new kernels have to satisfy a *sampling criterion*:

$$GBC_i \leq \lambda, \tag{4.5}$$

where λ acts as a *sampling threshold*, which is a binary separator deciding whether an FSE can be considered by the sampler as a *sample candidate* based on its bond confidence. Here we can find the first possible point for the fusion of top-down and

DOI:10.15774/PPKE.ITK.2013.005

4. ADAPTIVE EXTENSION

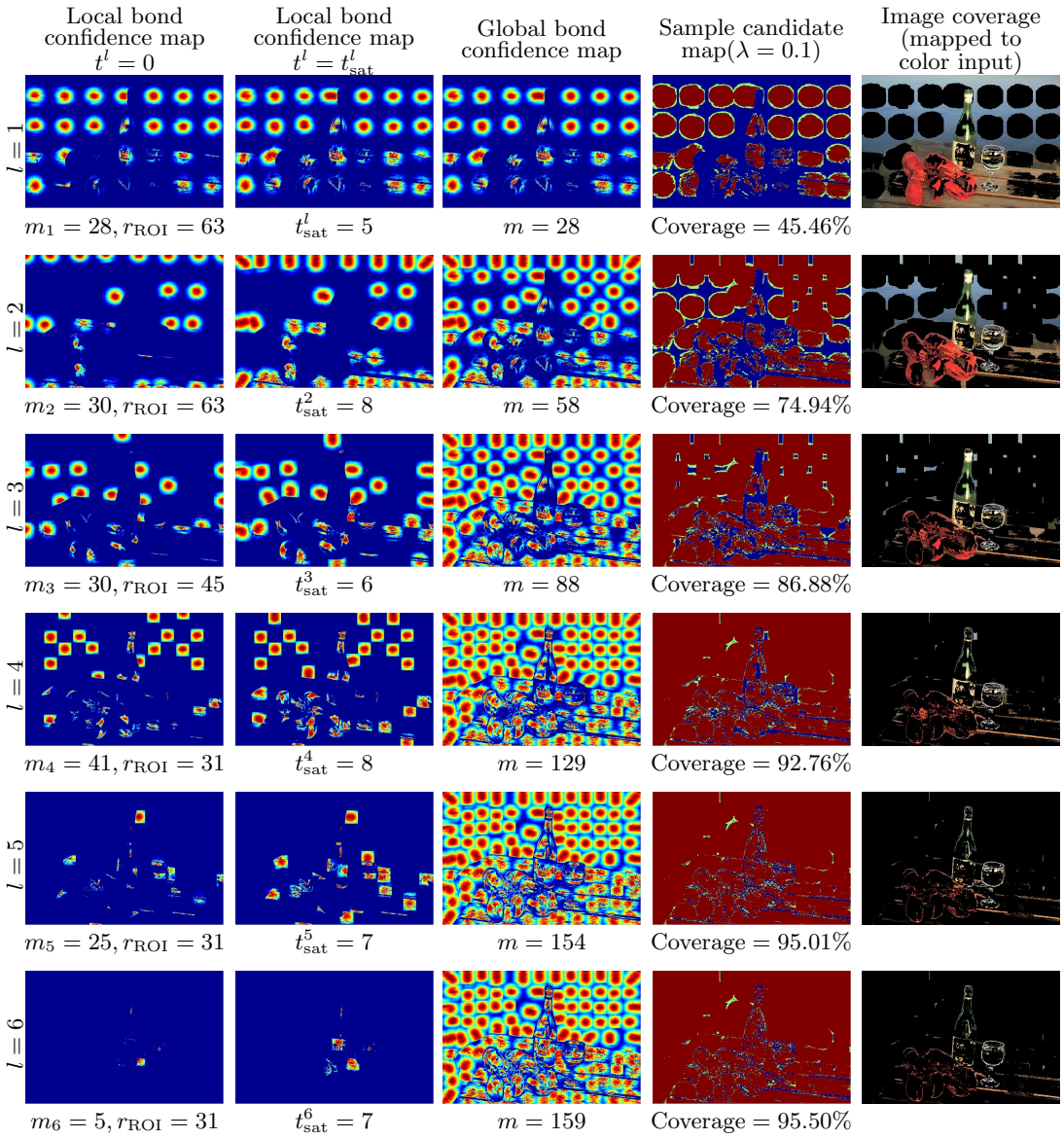


Figure 4.1: Iterations of the segmentation phase on a sample image. Rows represent consecutive loops. The first column displays bond confidences for the kernels initialized at iteration $t_l = 0$, where l is the respective loop number. The number of kernels initialized in the given loop is denoted by m_l, r_{ROI} indicates the ROI diameter of the spatial kernel. The second column displays the confidence values of the kernels initialized at iteration t^l_{sat}, which is the iteration where the saturation occurs. The third column depicts the graphical representation of the global bond confidence matrix with m being the cumulative number of kernels. In the case of the first three columns, a warmer color refers to a stronger bond between an FSE and its corresponding mode. Column four displays the information provided to the sampler through the state of the FSEs at t^l_{sat}: red, green and blue refers to strong, loose and no similarity to a mode, respectively. Coverage displays the percentage of FSEs already bound to a mode.

DOI:10.15774/PPKE.ITK.2013.005

bottom-up information, because the selection of the initial kernel positions from the candidates can be aided with *a priori* information if it is available for the task. (Since for now, the design of the discussed system aims at multipurpose applicability, it does not incorporate such semantic metadata.) The proposed sampling method was motivated by the main idea of the agglomerative clustering methods, i.e. that spatially nearby pixels are more likely to have similar color [76] than distant ones. Thus the sampler aims to select candidate FSEs that are the most distant in the spatial domain from the already bound FSEs, furthermore, the candidates should have a minimal distance of α from one another as well to minimize redundancy. In loop $l = 1$ sampling is done on an equidistant mesh (however, saliency-based pre-filtering [64] would also be possible). Beginning with loop no. 2, the first sample candidate is selected to be the spatially furthest FSE from non-candidate FSEs, then the following *sampling rules* are applied to select additional candidates, if possible:

1. the spatial distance between two sample candidates is at least $\alpha = \sqrt{2} \cdot r_{\mathrm{ROI}}$ to reduce the overlap (redundancy) between the mean shift kernels; and

2. the minimum distance between a sample candidate and any FSE with $GBC_i > \lambda$ is at least β, which is a user-selected distance threshold. (In the proposed system, β was selected to be 1% of the length of the smaller image axis but never larger than 5 pixels. A low value of λ causes more details to appear on the output, however, the computational demand will also be larger.)

In case the number of candidates satisfying these rules is smaller than m_l, the system can adaptively decrease r_{ROI} (i.e. not h_s), while simultaneously increasing m_l (observe Figure 4.1 for a practical example). According to my experiments, the FSEs representing pixels that are located on edges in the input image are tendentiously harder to cluster because their color intensity differs from the neighboring objects. The eccentricity of these edges is usually close to 1, such that it is computationally inefficient to apply wide kernels on them, therefore r_{ROI} is decreased, whereas the number of kernels m_l can be increased.

In the implementation, adjustment of these parameters is guided by the available device memory and the measured distance between the first selected sample and the closest non-candidate FSE. Algorithm 1 displays the sampling mechanism and algorithm 2 describes the adaptive ROI size adjustment.

DOI:10.15774/PPKE.ITK.2013.005

4. ADAPTIVE EXTENSION

Algorithm 1 Adaptive sampling

Require: min. sample candidate distance α, min. distance of a sample candidate and a non-candidate FSE β, distance matrix $\mathbf{DM}$, radius of kernel ROI r_{ROI}, number of kernels in the current loop m_l, kernel number multiplier η (1 by def.)

Ensure: kernel positions $\mathbf{KP}, r_{\mathrm{ROI}}, \eta$

$v \Leftarrow GetMaxVal(\mathbf{DM})$
$p \Leftarrow GetPos(v)$
$m_l \Leftarrow m_l\eta$
$\eta \Leftarrow 1$
if $\beta < v$ **then**
  $i \Leftarrow 0$
  $v \Leftarrow GetMaxVal(\mathbf{DM})$
  $p \Leftarrow GetPos(v)$
  $\alpha \Leftarrow \sqrt{2}r_{\mathrm{ROI}}$
  while $i \leq m_l$ **do**
    $i \Leftarrow i + 1$
    $\mathbf{DM}(SetNeighborhoodToZero(p, r_{\mathrm{ROI}})) \Leftarrow 0$
    $\mathbf{KP}_i \Leftarrow p$
    $v \Leftarrow GetMaxVal(\mathbf{DM})$
    $p \Leftarrow GetPos(v)$
    if $\alpha \geq v$ **then** {$0 < i < m_l$ candidates were sampled}
      break
    end if
  end while
  $[r_{\mathrm{ROI}}, \eta] \Leftarrow ROISizeAdjuster(v, r_{\mathrm{ROI}}, h_s, \mathbf{Dim}_I)$
else {No sample candidates were found}
  break
end if

Note: $\mathbf{DM}_i = \min(\|\mathbf{x}_{s,\mathrm{bound}} - \mathbf{x}_{s,i}\|)$

By applying the sampling criterion we obtain a map containing information not only about the current segmentation status of the FSEs, but also about the topographical extent of clustered and yet unclustered FSE regions. From this map, distances between candidate FSEs and non-candidate FSEs can easily be calculated using a distance transformation [78], such that sampling is performed adaptively with respect to the

DOI:10.15774/PPKE.ITK.2013.005

Algorithm 2 Adaptive ROI size adjustment (*ROISizeAdjuster*)

Require: maximum value of the DM v, radius of kernel ROI r_{ROI}, spatial kernel bandwidth h_s, spatial dimensions of the input image $\mathbf{Dim}_I$

Ensure: kernel number multiplier η, r_{ROI}

1: **if** $\min(\mathbf{Dim}_I/30) < v < \frac{3}{4}r_{\mathrm{ROI}}$ **then**
2: $\quad \eta \Leftarrow (\frac{4r_{\mathrm{ROI}}}{3v})^2$
3: $\quad r_{\mathrm{ROI}} \Leftarrow \frac{4}{3}v$
4: **end if**

image content in terms of the number of samples, the spatial position of the samples and the used radius of the applied Gaussian kernels.

4.2.2 Step 2—Mean Shift Iterations and Pixel Binding

As the next step, the mean shift algorithm is simultaneously executed on all m_l kernels (see Subsection 3.2.4). As the ROIs of certain kernels may overlap, in a parallel framework it is crucial to ensure mutual exclusion of the data accesses. For this reason, in each loop $l \in [1, L]$ a *local bond confidence matrix* is used. This is a temporary container which is used to register the highest bond confidence occurring for a given pixel in the current loop using the *pixel binding rule*:

$$\Upsilon_{K_l}(\chi_i) = \max_{j \in K_l}(\Upsilon_j(\chi_i)). \tag{4.6}$$

After all m_l kernels saturated, ψ_i mode positions are stored in the *local mode vector* (LMV).

However, in a parallel framework, kernels are handled in batches and a batch of kernels have to go through the same number of iterations, which brings a significant overhead, since all kernels have to be iterated until the last one saturates. In a GPGPU-based many-core realization, swapping of saturated kernels is computationally expensive due to the generated random memory accesses. To avoid this, the abridging approximation technique described in Subsection 3.2.5 was employed.

4.2.3 Step 3—Pixel-cluster Assignment

The final step of a loop covers four main subtasks.

DOI:10.15774/PPKE.ITK.2013.005

First, the GBC values are updated, such that $\forall i \in P_I$:

$$GBC_i = \max(GBC_i, \Upsilon_{K_l}(\chi_i)). \tag{4.7}$$

Second, for all elements in the GBC matrix that were modified by Equation 4.7, the PCM is set to point to the corresponding kernel. This way if in the current loop a new mode generated a higher bond confidence value for a given FSE, it is possible to rebind the corresponding pixel to it.

Third, those elements of the LMV that have FSEs bound to them are added to the GMV.

A natural termination point for the segmentation phase is when all elements of the feature space are clustered. However, in the case of real-life images, the input can contain a huge amount of thin gradients on the boundary of object and shaded spatial regions with a small extent, such that after a certain level, additional kernel initializations become highly suboptimal (see the progression of the coverage on the subfigures displaying image coverage in Figure 4.1).

For this reason, the fourth and final subtask of the pixel-cluster assignment is the examination of the *loop termination criterion* (LTC). The second possible entry point for top-down information injection is identified here: the selection of the LTC can be aided in case there are task-specific assumptions. In general, one of the most straightforward LTCs to choose is to check whether the number of unbound FSEs is under a certain threshold (e.g. 1%) and assign the unassigned elements of the PCM matrix to the mode that is closest to them (as discussed in Subsection 3.2.2). In the proposed framework, the segmentation phase is terminated in case no sample was retrieved by step 1, i.e. when the largest distance between a candidate sample and a non-candidate FSE was less than β. Consequently, objects having a color different from their neighborhood are detected in case the largest radius of their inscribed circle is at least β pixels, on the other hand objects with an incircle radius less then β pixels do not necessarily remain unbound. In case the LTC is not satisfied, a new loop is started, otherwise no more kernels will be initialized in addition to the existing $m = \sum_{l=1}^{L} m_l$ kernels.

Should the initial kernel positions be given *a priori* (e.g. by an oracle), then all kernels could be initialized in a single loop and the result would be exactly the same as with the method above. More formally, as a result of Equations 4.6 and 4.7, the GBC

DOI:10.15774/PPKE.ITK.2013.005

matrix will contain the maximal bond confidences for all FSEs:

$$\Upsilon(\chi_i) = \max_{j \in \bigcup_{l=1}^{L} K_l} (\Upsilon_j(\chi_i)), \tag{4.8}$$

and accordingly

$$\mathfrak{C}_i = \arg\max_j (\Upsilon_j(\chi_i)). \tag{4.9}$$

At the time instant the LTC is met, the FSEs can have three different states depending on their global bond confidence. An FSE_i can show

1. strong similarity to mode $\mathfrak{C}_i$, if $\lambda < \Upsilon(\chi_i)$;
2. loose similarity to mode $\mathfrak{C}_i$, if $0 < \Upsilon(\chi_i) \leq \lambda$;
3. no similarity to any mode, if $\Upsilon(\chi_i) = 0$.

By definition, strong similarity implies an existing pixel-cluster affiliation. An FSE χ_i with $0 < \Upsilon(\chi_i) \leq \lambda$ is bound to the mode to which it is loosely similar to. Remaining FSEs are bound to the same mode as the FSE with the smallest spatial distance from them, having a strong similarity.

The result of the sampling phase is an over-segmented output that is well-known [79, 80] and widely used [6, 81, 82] in the image processing community. The output cluster assigned to a pixel is defined by its PCM value, based on which the corresponding color information can be retrieved from the global mode vector.

The termination of the segmentation phase is followed by the merging phase.

4.3 Merging Phase

The second phase of the framework is merging, which finalizes the output of the segmentation by joining similar clusters. Cluster affinity can be a function of measured properties in some metrics, including the features used in the segmentation phase. Here is the third possible entry point for semantic information: the selection of these factors can be aided in case *a priori* knowledge is available for a segmentation task having specific priorities. The merging phase consists of recursive iterations, referred to as *rounds*, with each of them incorporating the following three steps:

DOI:10.15774/PPKE.ITK.2013.005

Step 1. Calculation of adjacency information (see Subsection 4.3.1);

Step 2. Similarity description (see Subsection 4.3.2);

Step 3. Cluster merging (see Subsection 4.3.3).

A new round is initiated as long as there are clusters to be merged subject to the applied similarity metric, such that the number of rounds (denoted by R) is determined by the complexity of the input image. The brief description of the steps of merging is given in the following.

4.3.1 Step 1—Calculation of Adjacency Information

Adjacency information is essentially a pre-filtering step. Clusters C_i and C_j are called *adjacent* when $\exists\{a,b\} \in P_I : a \neq b, \mathfrak{C}_a = C_i, \mathfrak{C}_b = C_j, \|\mathbf{x}_{s,a} - \mathbf{x}_{s,b}\| \leq \sqrt{2}$. This step is beneficial for two reasons. The first is that in the subsequent steps topographically non-adjacent clusters representing different objects with a similar color are not considered, only neighboring clusters are allowed to merge. The second is that since upcoming calculations of similarity are done for neighboring cluster pairs only, computational demand is reduced.

4.3.2 Step 2—Similarity Description

This is the key element of the merging phase, because the clusters to be joined will be selected based on the level of similarity calculated during this step. Due to the usage of real-life images, the algorithmic detection of perceptual homogeneity is not straightforward. Among several other causes, the main sources of the complexity of this task are luminance gradients caused by natural illumination, reflectance, and blurred color gradients caused by finite depth of field, because these phenomena alter the perceived color of pixels belonging to homogeneous regions. When considering a pair of clusters to be merged, the following properties were taken into account:

1. Color assigned to the cluster modes;

2. Average color of pixel sets of the input image residing on *neighborhood stripes*.

DOI:10.15774/PPKE.ITK.2013.005

A joinder may occur if the clusters are similar enough with respect to either of these properties. At first the distance of the color of the modes is measured using the linear combination of the Euclidean and an angular metric, which was motivated by the work of Wesolkowski [66] who used them for edge detection in color images.

Since the representation of a given color can substantially differ according to the color space used, the discrimination potential of the utilized metric highly depends on the space chosen. Consequently, the possible alternatives are discussed after the formal definition of each metric to justify the setting present in the proposed system.

The Euclidean distance is the metric most often used to measure similarity, while the vector angle proposed by Dony *et al.* [83] as a distance metric for colors is less known. When no further clusters can be merged using this combined metric, the algorithm tries to join the resulting bigger clusters based on the analysis of their corresponding neighborhood stripes.

The Euclidean distance of clusters C_i and C_j is calculated in the following way:

$$d_E(C_i, C_j) = \|\psi_{r,i} - \psi_{r,j}\|, \tag{4.10}$$

where ψ_r-s indicate the range information of modes belonging to adjacent clusters C_i and C_j respectively. It defines similarity through the magnitude of the vectors. When applied on the RGB space having three luminance-influenced channels, the metric describes differences using both intensity, hue, and saturation, but hue separation does not follow the human perception [66]. In case when chrominance-driven channels are present besides a luminance-related channel (i.e. when using e.g. the YCbCr or the Lab spaces), the metric characterizes differences of hue and intensity better.

The angular distance of clusters C_i and C_j with vector angle ϕ is calculated

$$d_A(C_i, C_j) = 1 - \sin\phi = \sqrt{1 - \left(\frac{\psi_{r,i}^T \psi_{r,j}}{\|\psi_{r,i}\|\|\psi_{r,j}\|}\right)^2}, \tag{4.11}$$

namely, similarity is defined through the direction of the vectors. In the case of using RGB color space, the value of the metric is sensitive to the differences in hue and saturation, but can tolerate changes in intensity (illumination) well that is a desirable characteristic in case real-life images are used. However, it is the value of the luminance-driven channel that affects the output of the angular metric the most, when it is used on color spaces with one luminance component and two chrominance channels with

DOI:10.15774/PPKE.ITK.2013.005

the hue also being a reliable descriptor. An important remark concerning the angular metric is that the discriminative power of the vector angle becomes unreliable in case any coordinate of either vector is small.

Since I am not aware of a work that provides numerical evaluation about the robustness of these distances on real-life images, it is somewhat hard to unambiguously isolate the strengths of the different metric-color space constellations in terms of saturation and intensity. For this reason, I build upon my own experience. Table 4.1 summarizes the favorable characteristics of the metrics used on different types of color spaces from the aspect of utilizing them on real-life images.

Table 4.1: Favorable characteristics of the different metrics used on real-life images, subject to the type of color space used. Boldface indicates the most beneficial properties for the proposed multipurpose system.

	RGB	One luminance channel, two chrominance channels
d_E	Intensity	**Intensity, (Hue)**
d_A	**Hue, Saturation, (Good intensity tolerance)**	Hue

The Euclidean distance is utilized on the Lab color space, because of its good capability of recognizing similarities especially in hue, which is heavily required for robust color similarity detection. However, the angular distance is applied in the RGB space, as this setting handles the merging of the similar-colored regions shaded by natural illumination well. Since the angular distance utilizes the global mode vector, color space conversion is done for $3m$ values, which is of low arithmetic demand.

The combined metric exploits the benefits of both the Euclidean and the angular distance having robust intensity description and robust hue description capabilities, respectively. To eliminate the weakness of the angular metric present at low intensities, it would be straightforward to use an intensity-driven tradeoff parameter that favors the Euclidean distance in case of such a color vector. Carron and Lambert [84] on

DOI:10.15774/PPKE.ITK.2013.005

the other hand suggested that color saturation is a more suitable tradeoff parameter. They argued, that when the saturation is low, intensity is less sensitive to noise than hue, thus the Euclidean distance characterizing intensity should be weighted more. On the contrary, at a high saturation, intensity is more sensitive to noise, therefore the angular distance—describing hue similarities—should be taken into account with a higher proportion.

For this reason, the two metrics were combined through a homotopy:

$$d_{AE} = \rho \cdot d_A + (1 - \rho) d_E, \tag{4.12}$$

which is driven by the *saturation parameter* ρ of the observed mode color value, which is calculated as

$$\rho = \sqrt{\mathsf{C}_1^2 + \mathsf{C}_2^2} \tag{4.13}$$

where C_1 and C_2 are chromatic channels that are obtained using the transformation proposed in [84]:

$$\begin{bmatrix} \mathsf{Y} \\ \mathsf{C}_1 \\ \mathsf{C}_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & -\frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} \mathsf{R} \\ \mathsf{G} \\ \mathsf{B} \end{bmatrix}. \tag{4.14}$$

Using the mode color in the combined metric is beneficial for two reasons. In addition to being computationally efficient due its compactness, its color represents a weighted average of the colors of the pixels belonging to the cluster considered. As a consequence of the second property, the mode color is especially useful in the case of surfaces with quasi-homogeneous illumination or with a fine texture. Unfortunately its descriptive power is limited from the aspect of merging in case of slowly evolving gradients. Consider Figure 4.2 that contains a sematic example of a soft gradient frequently present in an image containing natural illumination (e.g. in the sample images in Figure 4.6 and in B.1).

The effect of segmenting this gradient is similar to quantization in a sense that pseudo-linear intensity changes are estimated by a given number of discrete levels. Let us say that due to the color similarity measured by d_{AE}, adjacent clusters C_1 and C_2 and C_3 and C_4 got merged into cluster C_5 and C_6 respectively. As a consequence of the color assigned to the newly composed clusters being constructed from the mode color of their ancestors, the mode color of cluster C_5 will be brighter than the mode color of C_2, while the mode of cluster C_6 will be darker than C_3. Depending on the selected

DOI:10.15774/PPKE.ITK.2013.005

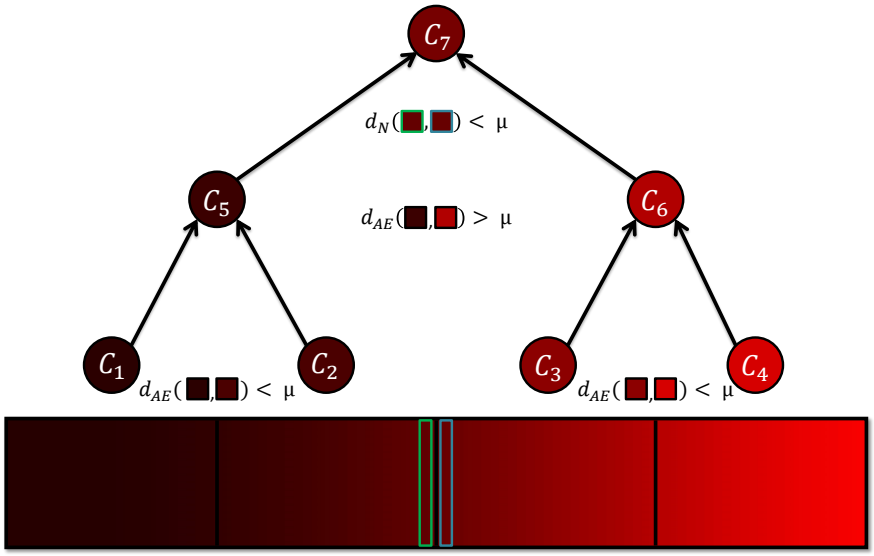


Figure 4.2: A sematic example showing the merging procedure of segmented clusters, encoding a slowly evolving gradient. The mode color $\psi_{r,i}$ of cluster C_i is represented by the color of the corresponding squares. d_{AE} and d_N indicate the color differences between the clusters subject to the joint angular-Euclidean metric and the neighborhood stripes, respectively. Despite belonging to the same region, the Euclidean distance of C_5 and C_6 exceeds the merging threshold μ, still the clusters get merged based on their similarity subject to d_N.

threshold of the applied metric, clusters C_5 and C_6 might not be considered similar during a subsequent similarity check, despite the fact that originally they encode parts of the same object, such that the area remains over-segmented.

The *neighborhood stripes*-based metric was designed to handle such cases. Neighborhood stripes of a cluster pair consist of the immediate neighbors in both clusters of the pixels belonging to the section of the border between the two clusters. Formally, let C_k and C_l denote two adjacent clusters. Then, for cluster C_k, the subset of pixels that reside on the *neighborhood stripe* of cluster C_l is denoted by P_{kl}^{δ} and is defined $P_{kl}^{\delta} = \{\mathsf{k} : \mathsf{k} \in C_k, \mathsf{k} \notin B_{kl}^{\delta}, \exists \mathsf{j} \in B_{kl}^{\delta}, \|\mathbf{x}_{s,\mathsf{k}} - \mathbf{x}_{s,\mathsf{j}}\| < 2\}$, where B_{kl}^{δ} refers to the boundary stripe of width δ and is defined as $B_{kl}^{\delta} = \{\mathsf{k} \in C_k : \exists \mathsf{l} \in C_l, \|\mathbf{x}_{s,\mathsf{k}} - \mathbf{x}_{s,\mathsf{l}}\| < \delta\}$.

DOI:10.15774/PPKE.ITK.2013.005

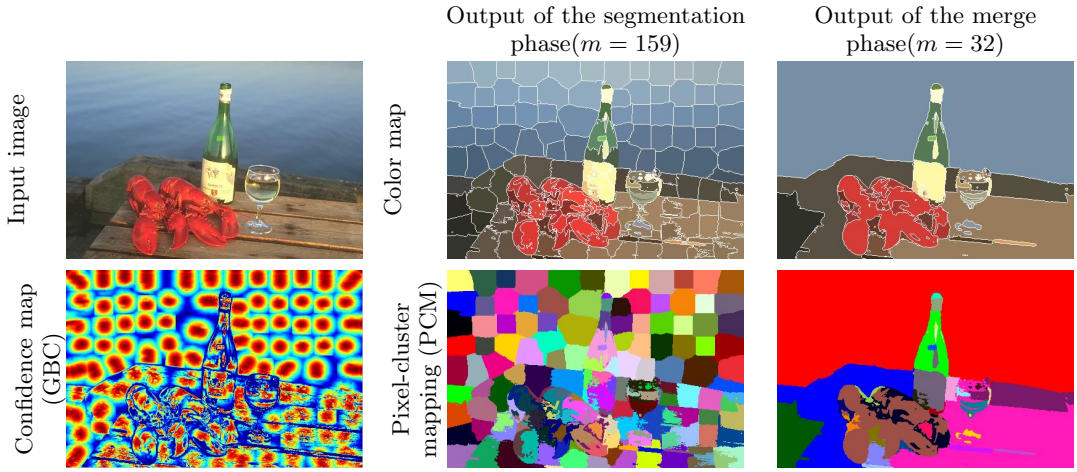


Figure 4.3: An example showing the results of the segmentation and merging phases, along with the most important matrices (in the form of map representations) used for the procedures. Based on the global bond confidence values (GBC, bottom left), the pixels of the input image (top left) are assigned to clusters (middle column) with the PCM matrix pointing to the modes defining their color. Merging (right column) is done based on the topography and color similarity of these clusters. The segmentation was done using the OPTIMAL parametrization.

Finally, the distance of the neighborhood stripes is given by

$$d_N(C_k, C_l) = \left\| \frac{\sum_{i \in P^{\delta}_{kl}} \mathbf{x}_{r,i}}{|P^{\delta}_{kl}|} - \frac{\sum_{j \in P^{\delta}_{lk}} \mathbf{x}_{r,j}}{|P^{\delta}_{lk}|} \right\|. \tag{4.15}$$

The greater the blur we expect in an image, the higher the δ parameter should be. All experiments were performed with $\delta = 2$.

4.3.3 Step 3—Cluster Merging

At the end of each merging round, cluster joining is done in the case of clusters that are *adequately similar*. Clusters C_i and C_j are adequately similar if they are adjacent, and

$$d_O(C_i, C_j) \leq \mu, \tag{4.16}$$

where $d_O \in \{d_{AE}, d_N\}$ is the metric utilized in the given round, and μ is the *merge distance threshold*.

Let us say that clusters C_i and C_j are adequately similar and are merged to form cluster $C_{(i;j)} = \{C_i \cup C_j\}$. In this case:

DOI:10.15774/PPKE.ITK.2013.005

1. the mode color information of cluster $C_{(i;j)}$ is calculated as

$$\psi_{r,(i;j)} = \frac{|C_i|}{|C_i \cup C_j|}\psi_{r,i} + \frac{|C_j|}{|C_i \cup C_j|}\psi_{r,j}, \tag{4.17}$$

2. the pixel-cluster mapping is refreshed, such that: $\forall a \in C_{(i;j)} : \mathfrak{C}_a = C_{(i;j)}$.

If no adequately close clusters were found in the current round, the merging procedure terminates and the framework returns the merged output, otherwise a new round is initiated. It was found that using a lower μ in case of the neighborhood stripes gives better results, consequently for this metric, the merge threshold was set to $\mu - 0.01$ in all of the experiments.

Figure 4.4 provides an analytical approach on displaying the influence of merge distance threshold on the output, while Figure 4.5 gives a systematical approach by showing an example for the practical effect of alternating the kernel bandwidths with parameter μ selected proportional to h_r. An important remark concerning the merging procedure is that although it reduces the over-segmentation, it cannot change the fundamental cluster structure, because it works on clusters resulted by the segmentation phase. However, if the framework is used for a particular segmentation task, the knowledge of the merger can easily be extended by task-specific rules, therefore this is the fourth identified entry point of top-down information.

As the conclusion of this section, the main steps and the graphical representation of the most important matrices used for the segmentation and merging phases are displayed in Figure 4.3.

4.4 Experimental Design

In practice, more than a year passed between the evaluation of the parallel framework and the adaptive-parallel extension. In the meantime not only the hardware environment changed, but a new generation of evaluation datasets had been released (these are the BSDS500 and the WIDB). These datasets (especially the BDSD500) offered a whole new set of conventional, uniformed metrics that are used as efficient tools for the assessment of the capabilities of algorithms, and thus for their comparison.

Hence the adaptive system was analyzed using a wider palette of metrics (see Subsection 2.5.4) and furthermore, a wider palette of analytical aspects (see Subsections

DOI:10.15774/PPKE.ITK.2013.005

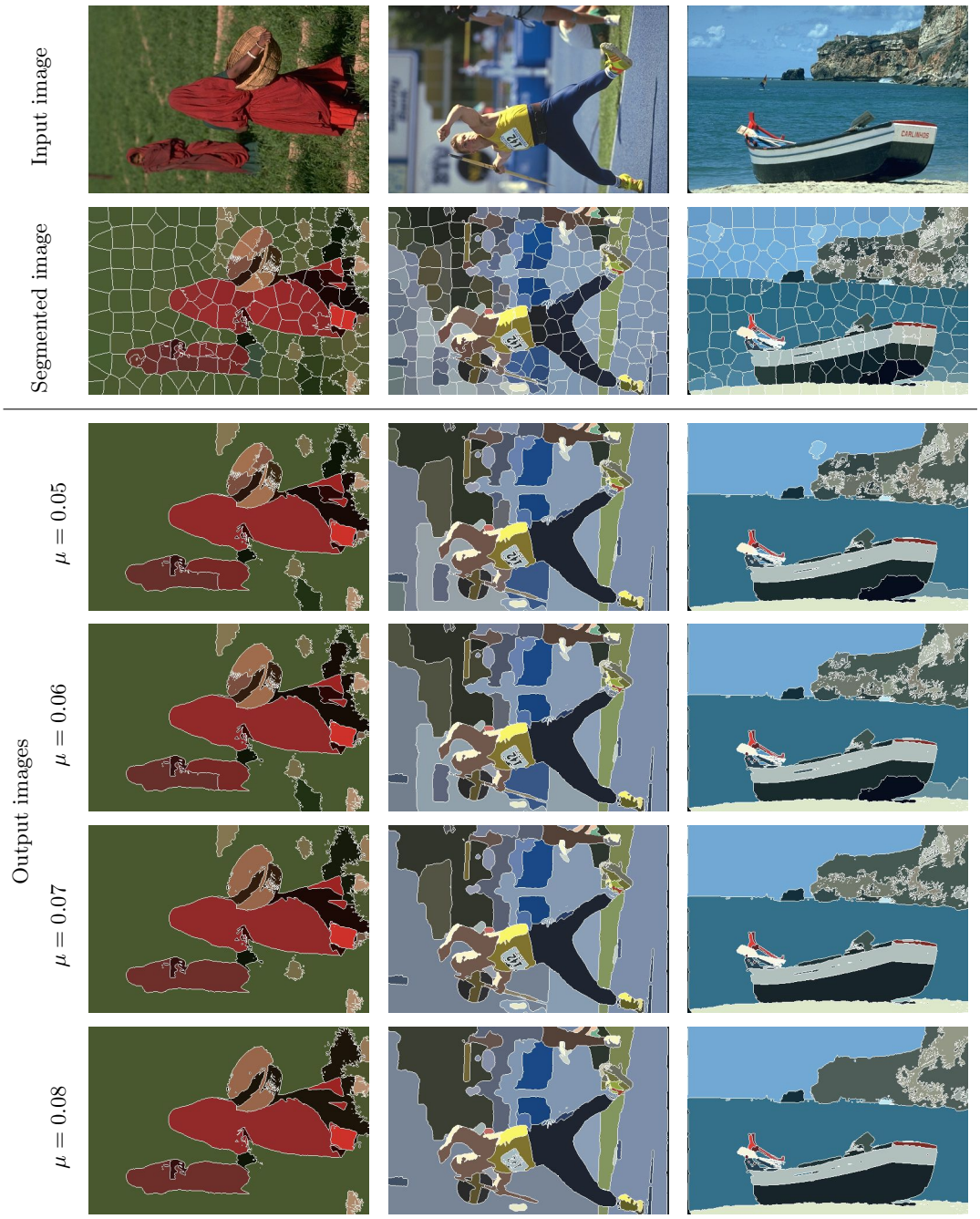


Figure 4.4: Segmentation outputs for different values of the merge distance threshold μ. Column 1 and 2 show the input image and the output of the segmentation phase, respectively. Merged images in columns 3 to 6 are obtained from the segmented image shown in column 2 by altering μ only.

DOI:10.15774/PPKE.ITK.2013.005

Figure 4.5: Sample output images for different spatial and range bandwidths, with the merge distance threshold $\mu = h_r + 0.02$.

2.5.1 and 2.5.2) into which the experience gained during the high resolution measurements made on the parallel framework was incorporated.

4.4.1 Hardware Specifications

The evaluation was performed on a PC equipped with 12GB of RAM, an Intel Xeon E5606 quad-core CPU clocked at 2.13GHz. The parallel framework used an nVidia 580GTX GPGPU containing 1.5GB of device memory and 512 stream processors (see Table 3.1).

4.4.2 Environmental Specifications

The environmental specifications used for the assessment of the adaptive framework were essentially the same as discussed in Subsection 3.3.3. A notable difference however

DOI:10.15774/PPKE.ITK.2013.005

is that all steps of the adaptive procedure utilized the Lab color space [77] unless it was indicated differently.

4.4.3 Experiments on Public Datasets

The training set of the BSDS300 was used to find generally eligible values for parameters h_s, h_r, μ and A. Output segmentations were generated for all 100 training images using all parameter combinations of $h_s = [0.02, 0.05], h_r = [0.02, 0.05], \mu = [0.05, 0.08]$ with step size being 0.01 and $A = \{0.6, 0.8, 1.0\}$. The output was evaluated for all 240 parametrizations using the provided metrics, finally, an ordered sequence of the goodness of a parameter setting was given by assigning the following weights to the values of the metrics: $w_{\mathcal{C}} = 0.3, w_{time} = 0.3, w_{PRI} = 0.175, w_F = 0.175, w_{VI} = 0.05$, that is, the main priority of parameter selection was the coverage metric and the running time demand.

Based on the evaluation the parameters (h_s, h_r, μ, A) were selected to be $(0.02, 0.05, 0.06, 0.6)$ respectively[5]. All results marked with OPTIMAL for the different assets of the Berkeley and Weizmann datasets were obtained using this parametrization. These results demonstrate the standard capabilities of the framework. Additionally, the best results achieved for the different quality metrics are reported under the label BEST.

By observing the OPTIMAL setting from the aspect of the mean shift calculation, the kernel bandwidths describe a mean shift kernel window that performs heavily blurring Gaussian low-pass filtering in the color domain within a small spatial area. The value of μ is very close to h_r, so they meet the $\mu = h_r$ recommendation appearing in [1]. The chosen abridging parameter ensures an optimal tradeoff between output quality and running time. The OPTIMAL setting offers advantages and disadvantages regarding output quality that are briefly summarized in the upcoming three paragraphs.

This setting works effectively in smoothing fine texture patterns such as grass, water, concrete or sand: such image regions are blurred due to the large color bandwidth, thus cluster fragmentation is relatively low in these areas. Similarly, most local shadings, which often appear due to natural lighting, are tolerated well. Due to the relatively

[5]Note: since the obtained bandwidth parameter values reside on the extrema of the evaluation intervals, further tests using a parameter domain enhanced with $h_r = 0.055, h_s = 0.015$ were carried out, but did not provide better overall results than the OPTIMAL setting.

DOI:10.15774/PPKE.ITK.2013.005

small spatial bandwidth the intra-cluster color difference between segments encoding slowly evolving gradients (see Subsection 4.3.2) is small, thus they can be concatenated during the merging phase.

Furthermore, the system can retrieve objects of small spatial dimensions in case they have a salient photometric representation. Such regions may contain important information, but in some cases they are not represented in ground truth images. Since most quality metrics punish over-segmentation at least as much as under-segmentation, the appearance of these additional details in the segmented image usually leads to lower output quality scores (see e.g. the third column of Figure 4.6). For this reason, judgement of the importance of detecting such small but salient segments in the BSDS is ambiguous.

On the other hand, many images of the BSDS contain foreground objects that have very similar color to one another and to the background, and are often separated by hardly visible contours. As the large color bandwidth of the OPTIMAL setting may eliminate such weak contours, cluster boundaries will not always match the borders of semantic objects in these cases. Efficient detection of such boundaries is possible only if additional cues (such as texture, depth information or the result of a multiscale analysis) are available [16], or high-level knowledge (either *a priori* or learned) can be integrated.

4.4.4 Experiments on High Resolution Images

To provide reference to the measurements concerning both the output quality and the running time, the EDISON system [60] was used to segment the high resolution image set. Segmentations were generated with all combinations of parameters $h_s = [20, 40]$ with a step size of $5, h_r = [7, 13]$ with a step size of 2 and $M = [100, 700]$ with a step size of 200 and the *high* speedup setting with the *speedup value* parameter left on the default value, 10.

A group of 15 participants were asked to select those segmentations from the resulted outputs, which they found to be perceptually the best. Next, for each participant, a list of the five parametrizations resulting in the highest number of selections in a descending order was composed. A score was assigned to each position on these lists, such that the i^{th} position was worth a score of $(6 - i)/5$. Finally the score for each parametrization was summarized and the parameter constellation with the highest score was selected to

DOI:10.15774/PPKE.ITK.2013.005

be the setting for the EDISON system to create the reference segmentation. Due to the utilized speedup parameter, this setting is referred to as the *high setting*. For the sake of a complete runtime analysis, the segmentation was done with the same (h_s, h_r, M) parametrization with no speedup applied. This parameter constellation is referred as the *naïve setting*.

Next, the framework was run on the high resolution set using the following parameter combinations: $h_s = 0.02$[6], $h_r = [0.02, 0.05]$ using step size 0.015 and $\mu = [0.05, 0.08]$ with step size 0.01. The abridging parameter was set to 0.6 to ensure fast computation. Again, the help of human participants was used to find the appropriate parametrization. In this case their task was to select images that were the most similar to the output of the reference segmentation. The construction of ordered lists and the assignment of scores to list elements were made in the same way as before, such that the parameter setting generating the most similar segmentation outputs subject to human perception was obtained.

Based on this evaluation, the best overall score for the proposed framework measured on the high resolution image set was resulted in using the $(h_s, h_r, \mu, A) = (0.02, 0.035, 0.06, 0.6)$ setting, thus it is referred to as setting HD-OPTIMAL, which was used for the subsequent analysis. This parametrization is very similar to the OPTIMAL setting, but in this case the range bandwidth parameter is smaller. The observation was that whereas the OPTIMAL parametrization provides a relatively under-segmented output that is desired by the BSDS, human perception leans towards preserving more details appearing only using a smaller h_r bandwidth.

After determining the settings for both frameworks, the analysis of running time results was performed on both the high resolution image set, and its three subsets formed based on the kappa-indices. The aim of the subset analysis is to show that the framework adaptively accelerates the segmentation procedure with respect to the amount of content in the image. The three classes created are as follows:

Class A: Images with $1 \leq \kappa < 2.5$, containing large, homogeneous surfaces and/or few objects and/or few details.

[6]The selection of system parameters used for the high resolution measurements was aided by the experience collected during the exhaustive measurements made on the BSDS300, e.g. it was found to be superfluous to try larger values for h_s, since the resolution is higher.

DOI:10.15774/PPKE.ITK.2013.005

Class B: Images with $2.5 \leq \kappa < 3.5$, containing homogeneous surfaces and/or a considerable number of objects and/or details.

Class C: Images with $3.5 \leq \kappa \leq 5$, containing plenty of details and a lot of elaborated objects.

4.5 Results

The framework was evaluated both for low and high resolution image sets separately due to the reasons discussed in Section 2.5.

The first batch details the results obtained using the public datasets and the metrics summarized above. The aim of these results is to provide complete, detailed, and comparable numerical quality assessment of the adaptive framework.

In the second batch the results obtained on the high resolution image set are examined. In this case the aim is to demonstrate that the framework provides an output quality similar to the publicly available reference implementation, but works faster. Furthermore, the running time demand of the system adapts to the content of the input image, and its relative standard deviation is much lower compared to the reference system.

An important note is that since the core of the framework remained the same, the results displayed by Figures 3.9 and 3.10 apply to the adaptive system as well, although the type of the CPU was different.

4.5.1 Evaluation on Public Datasets

Table 4.2 displays the results of the region measurements on the BSDS300 and on the BSDS500 datasets along with results of other mean shift-based approaches provided in the literature.

In [16], [36] and [85] the original Comaniciu and Meer mean shift method [1] was used for the related measurements. In [16] and [85] the applied parameter setting is not specified, whereas the authors of [36] run the evaluation using all combinations of $h_s = [7, 16], h_r = [3, 23]$ with regular step sizes on the dataset that contained images with the longer edge reduced equally to 320 pixels. They found the setting giving the

[7]Note: quality assessment was done using images with the longer edge reduced equally to 320 pixels.

DOI:10.15774/PPKE.ITK.2013.005

Table 4.2: Region benchmark results of different mean shift variants on the BSDS300 and the BSDS500. The displayed metrics are segmentation covering (Covering), probabilistic Rand index (PRI) and variation of information (VI). System parameters were static per image set for the Optimal Dataset Scale (ODS), or static per image for the Optimal Image Scale (OIS), whereas Best covering was obtained using any level from the segmentation hierarchy. BEST values represent the best values obtained using the whole test parameter space of h_s, h_r, μ, A, OPTIMAL values are obtained using fixed parameters optimized on the training set of the BSDS300. Measurements marked with an asterisk (*) were made using the system described in [60] with different parametrizations.

	BSDS300						BSDS500					
	Covering			PRI		VI	Covering			PRI		VI
	ODS	OIS	Best	ODS	OIS	ODS	OIS	ODS	OIS	Best	ODS	OIS	ODS	OIS
Kim* [85]	-	-	-	0.796	-	1.973	-	-	-	-	-	-	-	-
Arbeláez* [16]	0.54	0.58	0.66	0.78	0.80	1.83	1.63	0.54	0.58	0.66	0.79	0.81	1.85	1.64
BEST	**0.524**	**0.584**	**0.662**	**0.766**	**0.810**	**2.038**	**1.819**	**0.520**	**0.599**	**0.682**	**0.789**	**0.837**	**2.051**	**1.807**
OPTIMAL	**0.513**	-	-	**0.754**	-	**2.215**	-	**0.512**	-	-	**0.772**	-	**2.234**	-
Yang* [36][7]	-	-	-	0.755	-	2.477	-	-	-	-	-	-	-	-

DOI:10.15774/PPKE.ITK.2013.005

best output quality to be $(h_s, h_r) = (13, 19)$. The reason of the lower quality values compared to the ones in [16] might be the lower resolution used. As it is displayed in Table 4.2, despite the sampling scheme used, the output quality of the framework is comparable to the original mean shift in terms of the measured region metrics.

Table 4.3 displays the results of boundary measurements on the BSDS300 and on the BSDS500 datasets along with results of other mean shift-based approaches provided in the literature.

Table 4.3: Boundary benchmark results of different mean shift variants on the BSDS300 and the BSDS500. The F-measure values have been measured with two parametrizations, either static per image set for the Optimal Dataset Scale (ODS), or static per image for the Optimal Image Scale (OIS). AP denotes Average Precision. BEST values represent the best values obtained using the whole test parameter space of h_r, h_s, μ, A, OPTIMAL values are obtained using fixed parameters gained on the training set of the BSDS300. Measurements marked with an asterisk (*) were made using the system described in [60] with different parametrizations.

	BSDS300			BSDS500		
	F-measure		AP	F-measure		AP
	ODS	OIS		ODS	OIS	
Luo* [49]	0.673	-	-	-	-	-
Arbeláez* [16]	0.63	0.66	0.54	0.64	0.68	0.56
BEST	**0.614**	**0.625**	**0.525**	**0.624**	**0.650**	**0.541**
Paris [37]	0.61	-	-	-	-	-
OPTIMAL	**0.600**	**0.612**	**0.456**	**0.615**	**0.637**	**0.479**
Varga [86]	0.582	-	-	-	-	-
Kim* [87][8]	0.551	-	-	-	-	-

The result of [49] in Table 4.3 was obtained using the mean shift explained in [1] (reported parameters are: $h_s = \max(4, \min(height, width)/100), h_r = 5, M = 20, speedup = 20$) for the segmentation phase, and their own multiscale merging procedure. [87] also utilized the mean shift algorithm as discussed in [1] with parameters $(h_s, h_r, M) = (7, 6.5, 384)$ on images resized to 240×160 pixels to provide reference results to their work focusing on graph cut. Note that instead of the whole test set of

[8]Note: quality assessment was done using 60 hand-picked images with the longer edge reduced equally to 240 pixels.

DOI:10.15774/PPKE.ITK.2013.005

Table 4.4: F-measure results of different mean shift variants measured on the single-object Weizmann dataset. The foreground was fitted both with a single best-matching cluster provided by the segmenter and the union of multiple segments with an area considerably overlapping with it. System parameters were static per image set for the Optimal Dataset Scale (ODS), or static per image for the Optimal Image Scale (OIS). Results are displayed with the corresponding ξ values referring to the average number of segments. BEST refers to results obtained using the whole test parameter space of (h_s, h_r, μ, A), OPTIMAL refers to results obtained utilizing fixed parameters derived using the training set of the BSDS300. Note: both parametrizations used the color version of the images supplied with the database. The measurement marked with an asterisk (*) was made using the system described in [60].

	Single segment		Multiple segments			
	ODS	OIS	ODS	ξ	OIS	ξ
BEST	**0.682**	**0.781**	**0.914**	**25.91**	**0.944**	**18.510**
OPTIMAL	**0.618**	-	**0.859**	**10.820**	-	-
Alpert* [69]	0.57	-	0.88	12.08	-	-

the BSDS300, the quality assessments provided in this paper were made using 60 hand picked images from this set.

Figure 4.6 shows a few examples for the output of the proposed framework. The displayed images are from the BSDS500, the used setting is the OPTIMAL. Additional examples can be found in Appendix B.

Table 4.4 displays the results of the boundary measurements on the Weizmann dataset along with the mean shift scores published in [69].

The results in this paper were obtained using the mean shift as published in [1] with no parametrization discussed. The proposed framework was not retrained for the WIDB, such that the parametrization used for quality assessment was the same as in the case of the Berkeley datasets. However, it is noted that since in this dissertation the focus is on color segmentation, quality evaluation was performed on the color version of the database images that are also a part of the downloadable package. As the consequence of the presence of the additional chrominance information the discussed algorithm reached a slightly better ODS value in the single segment case. When applying the OPTIMAL parametrization in the multi-segment case, the ODS value of the F-measure is somewhat worse than the result published in [69], but the fragmentation

DOI:10.15774/PPKE.ITK.2013.005

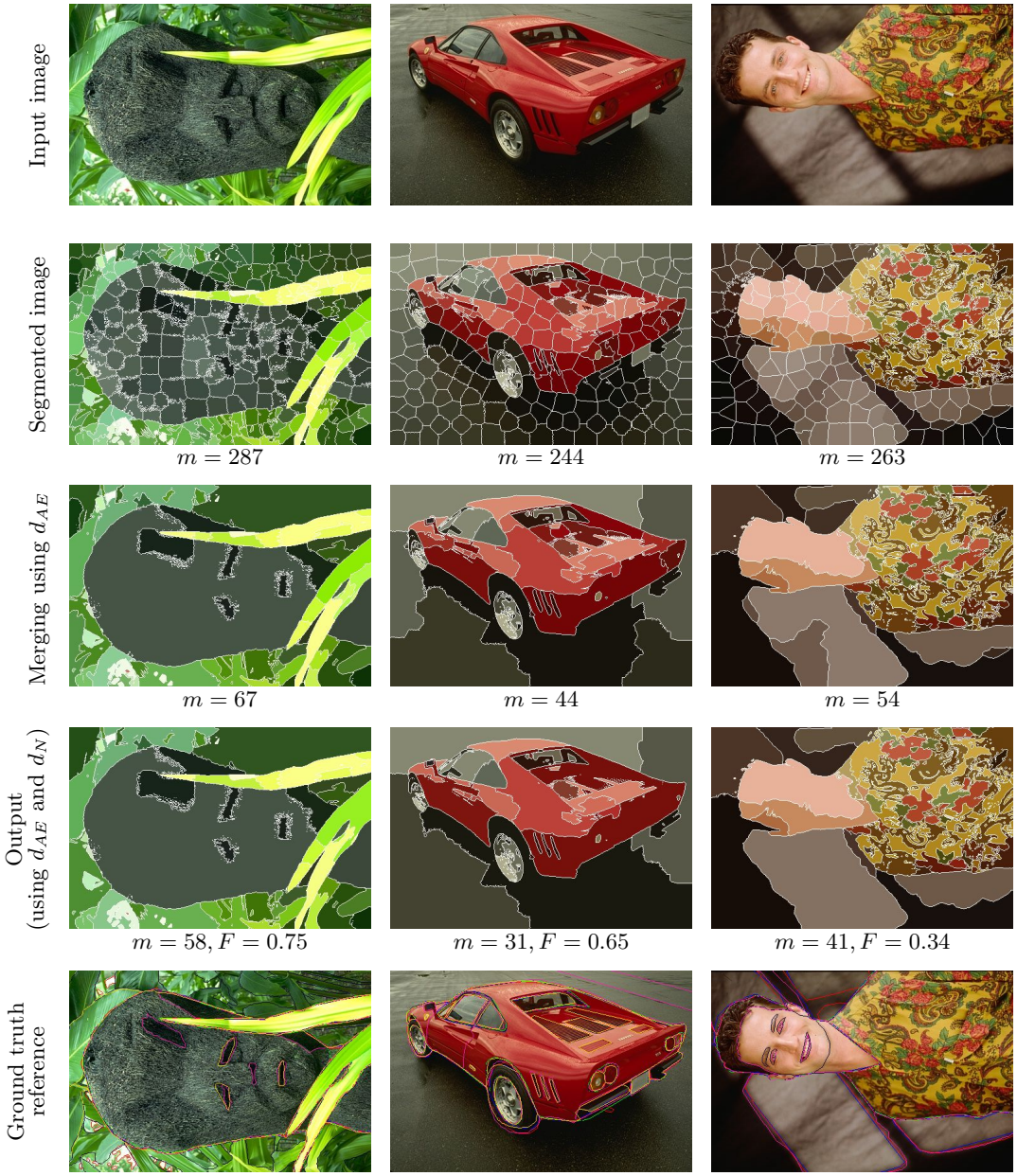


Figure 4.6: Segmentation examples from the test set of the BSDS500. The first row contains the input images, rows 2 to 4 show the results obtained at the end of certain stages of the procedure. The number of clusters is denoted by m, the F-measure of the segmentation output is denoted by F. Row 5 shows the boundaries of multiple ground truth segmentations provided as reference, with different colors. Segmentation was done using the OPTIMAL parametrization.

DOI:10.15774/PPKE.ITK.2013.005

in my case is smaller. OIS results are also provided along with the best results obtained for each measurement (displayed as BEST in Table 4.4). In the case of the multiple-segment evaluation, the only priority of parameter selection were the ODS and OIS values of the F-measure, consequently, the fragmentation is large.

4.5.2 Evaluation on High Resolution Images

Figure 4.7 and 4.8 show examples from the high resolution image set along with the output of the two phases of the proposed framework using the HD-OPTIMAL setting and the reference output obtained using the high setting. Additional examples can be found in Appendix C.

The clusters in the segmented images illustrate the behavior of the content-adaptive scheme: most regions that are quasi-homogeneous (e.g. sky, asphalt, grass, water surface etc.) are loosely sampled (characterized by large clusters), whereas crowded areas are clustered using many kernels, thus details are preserved (characterized by smaller clusters).

As the merged images show, the framework can handle most of the illumination-caused soft gradients, as such image regions are joined into the same cluster. Boundaries are accurate even for those objects that have many fine curves or tiny holes in them (such as foliage, or the details of windows on buildings). Such accurate boundaries are beneficial in case of e.g. an object detection task, when segmentation is instantly succeeded by classification, because only minor post-processing steps might be required to remove pixels not belonging to the object. The downside is that the presence of holes can lead to over-segmentation. To handle this problem, the algorithm of Comaniciu and Meer allows the user to select the smallest significant feature size (see Subsec 2.3.3). However, my experiences indicate that the proper selection of this parameter is very hard in the megapixel domain. The weakness of the proposed algorithm is that shadows, intensive shadings, and pixels that reside on, or close to object boundaries and thus are darker than their surround may cause unwanted over-segmentation. This is a problem commonly appearing in computer vision algorithms that work on natural images, and whereas it is well-studied in the case of image streams [88], the difficulty remains for single images, in which the integrated white condition [89] may not hold. Hue is considered to be a relatively reliable feature for this task, and the angular

DOI:10.15774/PPKE.ITK.2013.005

Figure 4.7: Segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries are marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

Figure 4.8: Segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

Table 4.5: Statistical results on the 103-item set of 10 megapixel images using the HD-OPTIMAL setting. As a reference the running times of the mean shift algorithm of Meer and Comaniciu [1, 60] are displayed using the *high speedup* setting and the *no speedup/naïve setting*. The speedup factor compares the running time of the proposed framework to the high setting of the reference system.

	Mean running time (s)			Average speedup
	The proposed system	Reference[9] (High setting)	Reference[9] (Naïve setting)	
Mean	**18.01**	320.01	23366.06	**18.58**
Std. dev./ (Relative)	**10.73** (59.59%)	277.46 (86.71%)	7385.67	**14.26**

distance separates differences in hue well. Unfortunately, the robustness of the angular metric becomes unreliable in the case of dark regions (see Subsection 4.3.2).

Table 4.5 displays the running time results measured on the high resolution image set consisting of 103 items.

As the table shows, it takes 18.01 seconds on average for the framework to segment a 10 megapixel image. This means, that the parallel system utilizing the many-core GPGPU completes the task 18.58 times faster than the publicly available reference implementation using the high speedup setting[9]. While the relative standard deviation of the reference system compared to its average running time is 86.71%, the same parameter is 59.59% in the case of the discussed framework, meaning that the system is more robust concerning the running time required for the segmentation task. For the sake of a complete comparison, Table 4.5 also contains the running time of the mean shift method with no speedup at all.

Measured on the whole high resolution image set, the correlation between the kappa-index and the number of kernels utilized per image by the proposed algorithm is 0.694, which indicates that there is a strong connection between what human image annotators pointed out, and what the framework indicated as image content.

[9] The precompiled version of the reference system (available from http://coewww.rutgers.edu/riul/research/code/EDISON/) was used that employs two discrete CPU cores.

DOI:10.15774/PPKE.ITK.2013.005

For additional investigation of the content-adaptive property, Table 4.6 displays the attributes of the three classes and the main numerical results measured using the proposed framework and the reference system.

Table 4.6: Statistical results on the three subsets containing 10 megapixel images. Subsets are based on human ratings of the complexity of image content, from images with only a few details/objects (*Class A*) to images with lots of objects/details (*Class C*). The average speedup compares the running times of the framework using the HD-OPTIMAL setting with the reference system using the *high setting*.

	Mean running time (s)			Average speedup	Average no. of kernels	No. of images
	The proposed system	Reference[9] (High setting)	Reference[9] (Naïve setting)			
Class A	**12.85**	204.22	23273.47	**15.54**	599.43	30
Class B	**14.84**	344.63	23371.93	**22.24**	674.38	45
Class C	**28.62**	404.49	23455.82	**15.96**	1101.18	28

The system achieves the highest relative speedup in the case of Class B that contains images with medium amount of information content. The reason for the speedup peak is that compared to the running times measured on Class A, the reference system requires 68.75% more time, while the proposed framework slows down by only 15.45%. The speedup gap gets smaller in the case of Class C that contains images with the most information, but the adaptive algorithm manages to operate almost 16 times as fast as the reference system using the high setting.

The correlation between the kappa-indices and the running times per image on the 103-element dataset was calculated. In the case of the reference system a correlation of 0.281 was measured, which indicates a weak connection, while in the case of the proposed framework the correlation is 0.676, which is almost as high as the correlation with the number of kernels. The numbers indicate that as the amount of image content grows, more and more kernels are used by the proposed framework to retrieve information. Paired with the running times, the results show that a more simple and mostly homogeneous image is segmented relatively quickly using only a few kernels, whereas the algorithm uses more kernels, and thus can retrieve more information, if the image contains many objects with fine details.

DOI:10.15774/PPKE.ITK.2013.005

4.6 Conclusion

In this chapter the adaptive extension of the parallel segmentation algorithm considered in Chapter 3 was discussed. The framework utilizes dynamical sampling that can determine both the number and the topographical position of the considered samples with respect to the content of the image. The main benefit of this adaptive system is that inhomogeneous image regions containing many details are densely sampled, thus such compressed information is kept in the output, while homogeneous regions are loosely sampled, such that the segmentation of these regions is very fast. Non-sampled pixels are assigned into clusters subject to a nonlinear similarity metric that considers both color similarity and spatial distance and is calculated without overhead. This approach makes the adaptive algorithm especially adequate for high resolution inputs.

In addition to the speedup due to the content-aware sampling, the parallel design of the proposed framework enables it to exploit the computation potential present in many-core processing units, thereby allowing for even faster processing.

The capabilities of the framework were assessed on multiple publicly available segmentation databases that use various metrics to measure segmentation quality. It was found that the output quality of the adaptive system is comparable to the existing mean shift-based segmenters. As I am aware of no conventional evaluation database in the high resolution image domain, several human subjects were asked to compare the output quality of the proposed system to the output of a publicly available reference implementation using a set of high resolution images. Based on this evaluation, the output quality remained comparable to the reference, but as numerical analysis demonstrated, the adaptive system provides output an order of magnitude faster. Additionally, human subjects were asked to rate the amount of the useful content in the high resolution images. Correlation analysis of the running time of the framework and the rates assigned to the images shows that the amount of speedup is proportional to the amount of details present in the images. My future work includes further investigation of a novel high resolution dataset that is suitable for the comparison of segmentation algorithms, moreover I plan to study the possibilities of constructing a metric that can measure image content.

The proposed system has been evaluated using generic, everyday images, however the modular design of the framework allows it to be enhanced with *a priori* information

DOI:10.15774/PPKE.ITK.2013.005

or task-specific rules. Highlighted points of knowledge injection include the following: addition of new heuristics and/or strategies in the sampling step (e.g. depending on certain properties of a region, such as its color, shape, or size, sampling can be more/less dense); selection of the loop termination criterion (e.g. loops can be terminated after a cluster with certain properties—color, shape, size, texture—is formed); rules applied in the merging phase (e.g. the merge threshold for colors with certain hue/intensity/value can be adjusted adaptively to be more strict/loose, or other color spaces and/or more advanced metrics can also be used).

DOI:10.15774/PPKE.ITK.2013.005

4. ADAPTIVE EXTENSION

DOI:10.15774/PPKE.ITK.2013.005

Chapter 5

Summary

5.1 Methods of Investigation

For the design of the algorithmic background, I relied on the literature available on kernel density estimation, sampling theory, Gaussian mixture modeling, color space theory, similarity metrics and parallel algorithmic design.

For the first batch of evaluations, I considered three major analytical aspects that are most frequently taken into account for an extensive assessment of a segmentation framework. These are the following: *running time demand* (the amount of time required to provide the clustered output from the input image); *output accuracy* (can be measured using several different metrics that compare the output of our system to a ground truth); and *physical resolution* (equivalent to the number of input image pixels).

As one of my primary aims was to provide results that are comparable to the ones published in the literature, I used publicly available, well-known datasets [16, 62, 69] for the analysis of output quality. These databases have the advantage of providing a huge variety of standardized metrics (including Segmentation Covering [90], Probabilistic Rand Index [68], Variation of Information [36], F-measure [91], Average Precision [92], and Fragmentation [69]) in a unified evaluation framework.

However, these benchmarks contain images of relatively low resolution, therefore their applicability to the other two mentioned aspects is limited. Since the results measured on the datasets referred to above can not be extended in a straightforward manner onto images of higher resolution, I compiled two additional high resolution image sets, both containing real-life images taken in natural environmental conditions

DOI:10.15774/PPKE.ITK.2013.005

and depicting objects of various scales. The first set consists of 15 high quality images in five different resolutions (see Table 3.2 for image specifications).

I used this set and a variety of general-purpose computing on graphics processing units (GPGPU) (see Table 3.1 for device specifications) to assess the running time and the algorithmic scaling of my framework. The evaluation made on this dataset confirmed my hypothesis that in the case of lossy, sampling-based segmentation algorithms, for a more complete evaluation a fourth analytical aspect, namely, the *image content*, should be taken into account. Consequently, I composed a second image set of 103 images with each having a resolution of 10 megapixels. In the case of the measurements made using this set, the main dimension of evaluation was not how the alternation of resolution influences the running time, but how the varying amount of content does.

My framework was implemented in MATLAB [93] and I used the Jacket toolbox developed by AccelerEyes [94]. This package enables the high level MATLAB code to run on the GPU, using a CUDA-based [95] back end. The advantage of the toolbox is that it provides the possibility of rapid prototyping, however, the initialization and fine adjustment of CUDA kernels remain hidden from the user. The statistical analysis was done using MATLAB and Microsoft Excel [96].

5.2 New Scientific Results

THESIS I. ***Parallelization and implementation of a bottom-up image segmentation algorithm on a many-core architecture.***

Most present-day photo sensors built into mainstream consumer cameras or even smartphones are capable of recording images of up to a dozen megapixels or more. In terms of computer vision tasks such as segmentation, image size is in most cases highly related to the running time of the algorithm. To maintain the same speed on increasingly large images, image processing algorithms have to run on increasingly powerful processing units. However, the traditional method of raising the core frequency to gain more speed—and computational throughput—has recently become limited due to the effect of high thermal dissipation, and the fact that semiconductor manufacturers are attacking atomic barriers in transistor design. For this reason, future trends of different types of processing elements—such as

DOI:10.15774/PPKE.ITK.2013.005

digital signal processors, field programmable gate arrays or GPGPUs—point towards the development of multi-core and many-core processors that can face the challenge of computational hunger by utilizing multiple processing units simultaneously. However, these architectures require new algorithms that are not trivial to bring to effect.

Related publications of the Author: [86, 97, 98].

I.1. I parallelized the mean shift segmentation algorithm, which this way became capable of exploiting the extra computational power offered by many-core platforms. I applied the method to several different general-purpose computing on graphics processing devices and showed that the acceleration resulting from the parallelized structure is proportional to the number of stream processors.

I designed an image segmentation framework that performs mean shift iterations on multiple kernels simultaneously. By implementing the system on a many-core architecture and assessing it on multiple devices having various number of stream processors I have experimentally proven that the parallel algorithm works significantly faster than its sequential version, furthermore, raising the number of processing units results in additional acceleration.

Figure 3.9 displays the speedup of the mean shift core of the system compared to a CPU (Intel Core i7-920 processor clocked at 2.66GHz).

I.2. Through the analysis of the overhead caused by the parallel scheme I showed that by the early termination of kernels requiring remarkably more computations than the average, one can gain significant acceleration, while at the same time, segmentation accuracy hardly drops according to the metrics generally used in the literature.

I found that it is not feasible to isolate saturated modes and replace them with new kernels in a "hot swap" way, due to the characteristics of block processing. I proposed a method (named *abridging*) to reduce the overhead caused by parallelization. I validated the relevance of the scheme through quality (see Figure 3.3) and running time evaluations made on the Berkeley Segmentation

DOI:10.15774/PPKE.ITK.2013.005

Dataset and Benchmark [62] and on a set of high resolution images (see Figure 3.6).

I.3. I created an efficient, parallel cluster merging algorithm that can decrease the over-segmentation of segmented images by using color and topographic information.

The concept of over-segmentation is well-known [79, 80] and widely used [6, 81, 82] in the image processing community. The main advantage of this scheme is that it makes the injection of both low and high-level information easy, thus the final cluster structure can be established using a set of rules that describe similarity with respect to the actual task. I have designed and implemented a parallel method for the computation of cluster neighborhood information and color similarity. Figure 4.6 shows a few examples of the results of the segmentation and merging procedures.

THESIS II. ***Adaptive, image content-based sampling method for nonparametric image segmentation.***

It is straightforward that the resolution of an image directly influences the running time and the output accuracy of a segmentation algorithm. But in case of lossy algorithms, the change in these two characteristics is not totally explained by the resolution because the distribution and amount of information in real-life images is very heterogeneous (see Figure 1.1), thus the results may depend on the characteristics of the input rather than the generic capabilities of the algorithm. From the aspect of computational complexity, the obvious priority here is to minimize the number of samples, but simultaneously we have to keep in mind that under-sampling introduces loss in image detail, whereas unnecessary over-sampling leads to computational overhead. To overcome these problems, I present the following contributions.

Related publication of the Author: [99].

II.1. I defined an implicitly calculated confidence value that is used as a heuristic for the adaptive sampling and at the same time is a sufficient guideline for the classification of image pixels.

DOI:10.15774/PPKE.ITK.2013.005

I gave a single-parameter system that registers the strength of the bond between a pixel and the mode of a cluster, based on their spatial distance and color similarity. This way each picture element is associated with the class having the most similar characteristics. The key element for both the sampling procedure and the voting algorithm is the *bond confidence value* that is calculated implicitly during the segmentation without introducing any overhead.

II.2. I developed a sampling scheme guided by the content of the image that adaptively chooses new samples at the appropriate location in the course of the segmentation. By evaluating my framework on both my high resolution dataset and on publicly available segmentation databases, I verified numerically that the quality indicators of the adaptive procedure are almost identical to those of the naïve method (employed on all pixels) subject to all prevalent metrics, but at the same time the computational demand is remarkably lower.

My segmentation algorithm utilizes adaptive sampling such that the sampling frequency is based on local properties of the image. Homogeneous image regions get clustered fast, initializing only a few large kernels, while spatially non-uniform regions, carrying fine details are processed using a larger number of smaller kernels that provide detailed information on them. While preserving the content of the image, this intelligent scheme reduces both the computational requirement and the memory demand, enabling the segmentation of high resolution images as well.

I showed via extensive output quality evaluation involving various metrics [36, 68, 69, 90, 91, 92] on multiple datasets [16, 62, 69] that despite the fact that my algorithm uses sampling, the segmentation quality it provides fits in well among the publicly available alternatives built on the mean shift segmentation procedure.

I performed running time measurements on a set of 103 high resolution images. To cope with the lack of ground truth required for quality assessment, human subjects were asked to select the parametrization with which the best output quality of the most popular, publicly available reference segmenter [60]

DOI:10.15774/PPKE.ITK.2013.005

was obtained, and the parametrization of my framework that results in the most similar output to that of the reference. Running time results measured on the dataset using these settings are shown in Table 4.5.

II.3. I created a measure to characterize the complexity of the content of images, and through high resolution time assessment and correlation analysis carried out between the running times and the amount of content indicated by my metric, I have empirically verified the adaptive behavior of my method, namely that the segmentation of images having less content is faster.

I defined a subjective, perception-based degree named the *kappa-index*. For a given image, it is calculated as the mean of ratings provided by human subjects, who are asked to assess the amount of useful content on a scale from 1 to 5, where 1 means a *"sparse image that contains only a few objects and large, homogenous regions"*, and 5 refers to a *"packed image having many identifiable details and rich information content"*. For each image in the 103-element high resolution dataset, the average rating of 15 participants was calculated and three subsets were formed based on the kappa-indices that represent the average amount of information in the images. Table 4.6 shows the running time results measured on the subsets.

Measured on the whole high resolution image set, the correlation between the kappa-index and the number of kernels utilized per image by my algorithm is 0.694, which indicates that there is a strong connection between what human image annotators pointed out, and what my framework indicated as image content.

5.3 Application of the Results

The methods presented in my dissertation can be applied as an intermediate step in several different assignments of pattern recognition, object detection, and high-level image understanding. The segmentation algorithm has already been applied successfully for two real-life scenarios.

DOI:10.15774/PPKE.ITK.2013.005

The first is the detection of crosswalks [98] that is a key task within the Bionic Eyeglass Project [100], an initiative that aims at giving personal assistance to the visually impaired in everyday tasks. The heart of this system is a portable device that, by analyzing multimodal information (such as visual data, GPS coordinates and environmental noises), is able to identify and recognize several interesting objects and patterns (e.g. clothes and their colors, pictograms, banknotes or bus numbers) and situations (e.g. incoming bus at the bus stop, or environments such as home/street/office). The information provided by this sensorial input can be injected into my algorithm in the form of merging rules, among others. For example, if the system can localize the position of the user via GPS coordinates, the rule set of the segmenter can be extended dynamically to compose objects (such as lamp posts or signs) with particular (color or shape) properties. Detection robustness can be enhanced this way, as many false positives are filtered out. As of now, smartphones offer not only a remarkable arsenal of sensors, but state of the art devices are also equipped with mobile GPUs that can be utilized by my parallel framework.

In the second case, my system was used in an early prototype of the Digital Holographic Microscope Project [101] that aims at developing an environment (hardware-software system) that is capable of autonomous water quality surveillance. To ensure robust detection, segmentation, and classification of different foreground objects (such as algae, pollens, or dust) from the background, the final version of the environment will use several different data sources including volumetric data (obtained via digital color holography) and material obtained with color and fluorescent microscopy. My framework was successfully used to segment the input provided in the form of a video frame sequence obtained using color light microscopy and fluorescent microscopy [102]. To ensure fast computation, the planned back-end system will be powered by specially designed many-core processors that again could be employed by my algorithm. Since the feature space my system works on is also a matter of selection, additional channels gained by various sensors is a possible way to further improve segmentation accuracy.

DOI:10.15774/PPKE.ITK.2013.005

5. SUMMARY

DOI:10.15774/PPKE.ITK.2013.005

References

[1] D. Comaniciu and P. Meer. **Mean Shift: A Robust Approach Toward Feature Space Analysis**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(5):603–619, 2002. 2, 11, 18, 19, 21, 24, 26, 39, 53, 89, 92, 94, 95, 100

[2] R. C. Gonzalez and R. E. Woods. *Digital Image Processing, Second Edition.* Prentice Hall, 2002. 10

[3] A. M. Treisman and G. Gelade. **A Feature-Integration Theory of Attention**. *Cognitive Psychology*, **12**(1):97–136, 1980. 10

[4] I. Biederman. **Recognition-by-components: A Theory of Human Image Understanding**. *Psychological Review*, **94**(2):115, 1987. 10

[5] J. Mutch and D. G. Lowe. **Object Class Recognition and Localization using Sparse Features with Limited Receptive Fields**. *International Journal of Computer Vision*, **80**(1):45–57, 2008. 10

[6] L. J. Li, R. Socher, and L. Fei-Fei. **Towards Total Scene Understanding: Classification, Annotation and Segmentation in an Automatic Framework**. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2036–2043. IEEE, 2009. 10, 79, 108

[7] A. K. Mishra, Y. Aloimonos, L. Cheong, and A. Kassim. **Active Visual Segmentation**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**(2):639–653, 2012. 10, 11, 29

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[8] E. Borenstein and J. Malik. **Shape Guided Object Segmentation**. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **1**, pages 969–976. IEEE Computer Society, 2006. 10

[9] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik. **Semantic Segmentation using Regions and Parts**. *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 10

[10] R. Zhao and W. I. Grosky. **Bridging the Semantic Gap in Image Retrieval**. *Distributed Multimedia Databases: Techniques and Applications*, pages 14–36, 2001. 10

[11] X. M. He, R. S. Zemel, and D. Ray. **Learning and Incorporating Top-Down Cues in Image Segmentation**. In *Proceedings of the 2006 European Conference on Computer Vision*, pages I: 338–351, 2006. 10

[12] L. Dong and E. Izquierdo. **A Biologically Inspired System for Classification of Natural Images**. *IEEE Transactions on Circuits and Systems for Video Technology*, **17**(5):590–603, May 2007. 10

[13] E. Borenstein and S. Ullman. **Combined Top-down/Bottom-up Segmentation**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2109–2125, 2007. 10

[14] F. Shahbaz Khan, J. van de Weijer, and M. Vanrell. **Top-down Color Attention for Object Recognition**. In *Proceedings of the 12th IEEE International Conference on Computer Vision and Pattern Recognition*, pages 979–986. IEEE, 2009. 10

[15] T. Malisiewicz and A. A. Efros. **Improving Spatial Support for Objects via Multiple Segmentations**. In *Proceedings of the 2007 British Machine Vision Conference*. The British Machine Vision Association, 2007. 11

[16] P. Arbeláez, M. Maire, C. Fowlkes, and M. Malik. **Contour Detection and Hierarchical Image Segmentation**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(5):898–916, 2011. 11, 13, 17, 29, 33, 37, 90, 92, 93, 94, 105, 109

DOI:10.15774/PPKE.ITK.2013.005

[17] J. Shi and J. Malik. **Normalized Cuts and Image Segmentation**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(8):888–905, 2000. 11, 14

[18] P. F. Felzenszwalb and D. P. Huttenlocher. **Efficient Graph-Based Image Segmentation**. *International Journal of Computer Vision*, **59**(2):167–181, September 2004. 11, 13

[19] M. B. Salah, A. Mitiche, and I. B. Ayed. **Multiregion Image Segmentation by Parametric Kernel Graph Cuts**. *IEEE Transactions on Image Processing*, **20**(2):545–557, 2011. 11, 14

[20] C. Nikou, N. P. Galatsanos, and A. C. Likas. **A Class-adaptive Spatially Variant Mixture Model for Image Segmentation**. *IEEE Transactions on Image Processing*, **16**(4):1121–1130, 2007. 11, 16

[21] X. Ren and J. Malik. **Learning a Classification Model for Segmentation**. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 10–17. IEEE, 2003. 11, 15

[22] A. Hinneburg and D. A. Keim. **Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering**. In *Proceedings of the 25th International Conference on Very Large Data Bases*, page 517. Morgan Kaufmann Publishers Inc., 1999. 12

[23] S. X. Yu. **Segmentation Induced by Scale Invariance**. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **1**, pages 444–451. IEEE Computer Society, 2005. 13

[24] D. R. Martin, C. C. Fowlkes, and J. Malik. **Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**(5):530–549, 2004. 13, 54

[25] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. **Using Contours to Detect and Localize Junctions in Natural Images**. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 13

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[26] B. C. Catanzaro, B. Su, N. Sundaram, Y. Lee, M. Murphy, and K. Keutzer. **Efficient, High-quality Image Contour Detection**. In *Proceedings of the 12th IEEE International Conference on Computer Vision*, pages 2381–2388. IEEE, 2009. 13

[27] J. Wassenberg, W. Middelmann, and P. Sanders. **An Efficient Parallel Algorithm for Graph-based Image Segmentation**. In *Computer Analysis of Images and Patterns*, pages 1003–1010. Springer, 2009. 13

[28] P. Strandmark and F. Kahl. **Parallel and Distributed Graph Cuts by Dual Decomposition**. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2085–2092. IEEE, 2010. 14

[29] G. L. Miller and D. A. Tolliver. **Graph Partitioning by Spectral Rounding: Applications in Image Segmentation and Clustering**. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **1**, pages 1053–1060. IEEE, 2006. 15

[30] W. Y. Chen, Y. Song, H. Bai, C. J. Lin, and E. Y. Chang. **Parallel Spectral Clustering in Distributed Systems**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(3):568–586, 2011. 15

[31] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. **Superpixel Lattices**. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 15

[32] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. **TurboPixels: Fast Superpixels Using Geometric Flows**. *IEEE Transactios on Pattern Analysis and Machine Intelligence*, **31**(12):2290–2297, December 2009. 16

[33] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. **SLIC Superpixels**. Technical report, EPFL Technical Report, 2010. 16

[34] C. Y. Ren and I. Reid. **gSLIC: A Real-time Implementation of SLIC Superpixel Segmentation**. Technical report, University of Oxford, Department of Engineering Science, 2011. 16

DOI:10.15774/PPKE.ITK.2013.005

[35] C. Nikou, A. C. Likas, and N. P. Galatsanos. **A Bayesian Framework for Image Segmentation with Spatially Varying Mixtures**. *IEEE Transactions on Image Processing*, **19**(9):2278–2289, 2010. 16

[36] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. **Unsupervised Segmentation of Natural Images Via Lossy Data Compression**. *Computer Vision and Image Understanding*, **110**(2):212–225, 2008. 16, 92, 93, 105, 109

[37] S. Paris and F. Durand. **A Topological Approach to Hierarchical Segmentation using Mean Shift**. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 17, 25, 39, 43, 94

[38] J. A. Hartigan and M. A. Wong. **Algorithm AS 136: A k-means Clustering Algorithm**. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **28**(1):100–108, 1979. 18

[39] S. H. Roosta. *Parallel Processing and Parallel Algorithms: Theory and Computation*. Springer, 1999. ISBN-10: 0387987169, ISBN-13: 978-0387987163. 18

[40] Y. Cheng. **Mean Shift, Mode Seeking, and Clustering**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(8):790–799, 1995. 18, 19, 23, 47

[41] K. Fukunaga and L. Hostetler. **The Estimation of the Gradient of a Density function, with Applications in Pattern Recognition**. *IEEE Transactions on Information Theory*, **21**(1):32–40, 1975. 18

[42] D. DeMenthon. **Spatio-temporal Segmentation of Video by Hierarchical Mean Shift Analysis**. In *Language*, **2**, page 1. Center for Automation Research, University of Maryland, College Park, 2002. 23, 39

[43] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis. **Improved Fast Gauss Transform and Efficient Kernel Density Estimation**. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 664–671, 2003. 23, 39

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[44] C. Yang, R. Duraiswami, D. DeMenthon, and L. Davis. **Mean-shift Analysis Using Quasi-Newton Methods**. In *Proceedings of the International Conference on Image Processing*, pages 447–450, 2003. 23, 39

[45] B. Georgescu, I. Shimshoni, and P. Meer. **Mean Shift Based Clustering in High Dimensions: A Texture Classification Example**. In *Proceedings of the 9th IEEE International Computer Vision Conference*, pages 456–463, 2003. 23, 39

[46] M. A. Carreira-Perpiñán. **Acceleration Strategies for Gaussian Mean-Shift Image Segmentation**. In *Proceedings of the 2006 IEEE Computer Society Conference Computer Vision and Pattern Recognition*, **1**, pages 1160–1167, 2006. 24, 39, 43

[47] M. A. Carreira-Perpiñán. **Gaussian Mean-Shift is an EM Algorithm**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(5):767–776, 2007. 24, 39

[48] M. A. Carreira-Perpiñán. **Fast Nonparametric Clustering with Gaussian Blurring Mean-shift**. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 153–160, New York, NY, USA, 2006. ACM. 24, 39

[49] Q. Luo and T. M. Khoshgoftaar. **Unsupervised Multiscale Color Image Segmentation based on MDL Principle**. *IEEE Transactions on Image Processing*, **15**(9):2755–2761, 2006. 24, 39, 94

[50] D. Comaniciu. **An Algorithm for Data-driven Bandwidth Selection**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(2):281–288, 2003. 24, 39

[51] P. Wang, D. Lee, A. G. Gray, and J. M. Rehg. **Fast Mean Shift with Accurate and Stable Convergence**. *Journal of Machine Learning Research - Proceedings Track*, **2**:604–611, 2007. 24, 39

[52] J. Wang, B. Thiesson, Y. Xu, and M. Cohen. **Image and Video Segmentation by Anisotropic Kernel Mean Shift**. In T. Pajdla and

DOI:10.15774/PPKE.ITK.2013.005

J. Matas, editors, *Proceedings of the 2004 European Conference on Computer Vision*, **3022** of *Lecture Notes in Computer Science*, pages 238–249. Springer Berlin, Heidelberg, 2004. 25

[53] H. Guo, P. Guo, and H. Lu. **A Fast Mean Shift Procedure with New Iteration Strategy and Re-sampling**. In *Proceedings of the 2006 IEEE International Conference on Systems, Man and Cybernetics*, **3**, pages 2385–2389, 2006. 25, 39, 43

[54] A. Pooransingh, C. A. Radix, and A. Kokaram. **The Path Assigned Mean Shift Algorithm: A New Fast Mean Shift Implementation for Colour Image Segmentation**. In *Proceedings of the 15th IEEE International Conference on Image Processing*, pages 597–600. IEEE, 2008. 25, 39, 71

[55] F. Zhou, Y. Zhao, and K. Ma. **Parallel Mean Shift for Interactive Volume Segmentation**. In F. Wang, P. Yan, K. Suzuki, and D. Shen, editors, *Machine Learning in Medical Imaging*, **6357** of *Lecture Notes in Computer Science*, pages 67–75. Springer, 2010. 25, 39, 43, 64

[56] C. Xiao and M. Liu. **Efficient Mean-shift Clustering Using Gaussian KD-Tree**. *Computer Graphics Forum*, **29**(7):2065–2073, 2010. 26, 39, 64

[57] D. Freedman and P. Kisilev. **Fast Mean Shift by Compact Density Representation**. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1818–1825, 2009. 26, 39

[58] D. Freedman and P. Kisilev. **KDE Paring and a Faster Mean Shift Algorithm**. *SIAM Journal on Imaging Sciences*, **3**(4):878–903, 2010. 26, 39, 43

[59] K. Zhang and J. T. Kwok. **Simplifying Mixture Models Through Function Approximation**. *IEEE Transactions on Neural Networks*, **21**(4):644–658, 2010. 26, 39, 64

[60] C. M. Christoudias, B. Georgescu, and P. Meer. **Synergism in Low Level Vision**. In *Proceedings of the 16th IEEE International Conference on Pattern Recognition*, **4**, pages 150–155. IEEE, 2002. 26, 90, 93, 94, 95, 100, 109

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[61] H. Zhang, J. E. Fritts, and S. A. Goldman. **Image Segmentation Evaluation: A Survey of Unsupervised Methods**. *Computer Vision and Image Understanding*, **110**(2):260–280, 2008. 27, 34

[62] D. R. Martin, C. C. Fowlkes, D. Tal, and J. Malik. **A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics**. In *Proceedings of the 8th IEEE International Conference on Computer Vision*, **2**, pages 416–423, July 2001. 28, 29, 33, 48, 105, 108, 109

[63] S. Bagon, O. Boiman, and M. Irani. **What is a Good Image Segment? A Unified Approach to Segment Extraction**. *Proceedings of the 2008 European Conference on Computer Vision*, pages 30–44, 2008. 29

[64] T. Kadir and M. Brady. **Saliency, Scale and Image Description**. *International Journal of Computer Vision*, **45**(2):83–105, 2001. 29, 30, 31, 72, 75

[65] S. Khan and M. Shah. **Object Based Segmentation of Video Using Color, Motion and Spatial Information**. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **2**, pages II–746. IEEE, 2001. 29

[66] S. B. Wesolkowski. *Color Image Edge Detection and Segmentation: A Comparison of the Vector Angle and the Euclidean Distance Color Similarity Measures*. PhD thesis, University of Waterloo, 1999. 29, 81

[67] X. Jiang, C. Marti, C. Irniger, and H. Bunke. **Distance Measures for Image Segmentation Evaluation**. *EURASIP Journal on Applied Signal Processing*, **2006**:209–209, 2006. 29

[68] R. Unnikrishnan, C. Pantofaru, and M. Hebert. **Toward Objective Evaluation of Image Segmentation Algorithms**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(6):929–944, 2007. 29, 105, 109

[69] S. Alpert, M. Galun, A. Brandt, and R. Basri. **Image Segmentation by Probabilistic Bottom-Up Aggregation and Cue Integration**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**:315–327, February 2012. 29, 33, 38, 95, 105, 109

DOI:10.15774/PPKE.ITK.2013.005

[70] K. Yanai and K. Barnard. **Image Region Entropy: A Measure of Visualness of Web Images Associated with One Concept**. In *Proceedings of the 13th Annual ACM international Conference on Multimedia*, pages 419–422. ACM, 2005. 31

[71] B. Moghaddam, H. Biermann, and D. Margaritis. **Defining Image Content with Multiple Regions-of-interest**. In *Proceedings of the 1999 IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 89–93. IEEE, 1999. 32

[72] N. Abbadeni, D. Zhou, and S. Wang. **Computational Measures Corresponding to Perceptual Textural Features**. In *Proceedings of the 2000 International Conference on Image Processing*, **3**, pages 897–900. IEEE, 2000. 32

[73] S. Alpert, M. Galun, R. Basri, and A. Brandt. **Image Segmentation by Probabilistic Bottom-Up Aggregation and Cue Integration**. In *Proceedings of the of the 2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, June 2007. 33

[74] H. El-Rewini and M. Abd-El-Barr. *Advanced Computer Architecture and Parallel Processing (Wiley Series on Parallel and Distributed Computing)*. Wiley, 2005. ISBN-10: 0471467405, ISBN-13: 978-0471467403. 42

[75] C. Jia, W. Xiaojun, and C. Rong. **Parallel Processing for Accelerated Mean Shift Algorithm**. *Journal of Computer Aided Design and Computer Graphics*, (3):461–466, 2010. 43, 64

[76] C. Tomasi and R. Manduchi. **Bilateral Filtering for Gray and Color Images**. In *Proceedings of the 6th International Conference on Computer Vision*, pages 839–846. IEEE, 1998. 70, 75

[77] K. Bitsakos, C. Fermüller, and Y. Aloimonos. **An Experimental Study of Color-based Segmentation Algorithms Based on the Mean-shift Concept**. *Proceedings of the 2010 European Conference on Computer Vision*, pages 506–519, 2010. 70, 89

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[78] C. R. Maurer Jr, R. Qi, and V. Raghavan. **A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(2):265–270, 2003. 76

[79] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher. **An Experimental Comparison of Range Image Segmentation Algorithms**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(7):673–689, 1996. 79, 108

[80] A. Duarte, Á. Sánchez, F. Fernández, and A. S. Montemayor. **Improving Image Segmentation Quality Through Effective Region Merging using a Hierarchical Social Metaheuristic**. *Pattern Recognition Letters*, **27**(11):1239–1251, 2006. 79, 108

[81] S. Beucher. **Watershed, Hierarchical Segmentation and Waterfall Algorithm**. *Mathematical Morphology and its Applications to Image Processing*, pages 69–76, 1994. 79, 108

[82] F. Y. Shih and S. Cheng. **Automatic Seeded Tegion Growing for Color Image Segmentation**. *Image and Vision Computing*, **23**(10):877–886, 2005. 79, 108

[83] R. D. Dony and S. Wesolkowski. **Edge Detection on Color Images Using RGB Vector Angles**. In *Proceedings of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering*, **2**, pages 687–692. IEEE, 1999. 81

[84] T. Carron and P. Lambert. **Color Edge Detector Using Jointly Hue, Saturation and Intensity**. In *Proceedings of the 1994 IEEE International Conference on Image Processing*, **3**, pages 977–981. IEEE, 1994. 82, 83

[85] T. H. Kim, K. M. Lee, and S. U. Lee. **Learning Full Pairwise Affinities for Spectral Segmentation**. In *Proceedings of the of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2101–2108. IEEE, 2010. 92, 93

DOI:10.15774/PPKE.ITK.2013.005

[86] B. Varga and K. Karacs. **High-resolution Image Segmentation Using Fully Parallel Mean Shift**. *EURASIP Journal on Advances in Signal Processing*, **2011**(1):111, 2011. 94, 107

[87] J. S. Kim and K. S. Hong. **A new Graph Cut-based Multiple Active Contour Algorithm without Initial Contours and Seed Points**. *Machine Vision and Applications*, **19**(3):181–193, 2008. 94

[88] A. Prati, I. Mikic, M. M. Trivedi, and R. Cucchiara. **Detecting Moving Shadows: Algorithms and Evaluation**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(7), 2003. 97

[89] H. D. Cheng, X. H. Jiang, Y. Sun, and J. Wang. **Color Image Segmentation: Advances and Prospects**. *Pattern Recognition*, **34**(12):2259–2281, 2001. 97

[90] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. **PASCAL 2008 Results**. 2008. 105, 109

[91] N. Chinchor and B. Sundheim. **MUC-5 Evaluation Metrics**. In *Proceedings of the 5th Conference on Message Understanding*, pages 69–78. Association for Computational Linguistics, 1993. 105, 109

[92] B. Bartell, G. Cottrell, and R. Belew. **Optimizing Parameters in a Ranked Retrieval System Using Multi-query Relevance Feedback**. In *Proceedings of the 1994 Symposium on Document Analysis and Information Retrieval (SDAIR)*. Citeseer, 1994. 105, 109

[93] The MathWorks Inc. **MATLAB**. Accessed June, 2012. 106

[94] AccelerEyes LLC. **Jacket**. Accessed June, 2012. 106

[95] NVIDIA Corporation. **CUDA**. Accessed August, 2012. 106

[96] The Microsoft Corporation. **Microsoft Excel 2010**. Accessed June, 2012. 106

DOI:10.15774/PPKE.ITK.2013.005

REFERENCES

[97] B. Varga and K. Karacs. **GPGPU Accelerated Scene Segmentation Using Nonparametric Clustering**. In *Proceedings of the 2010 International Symposium on Nonlinear Theory and its Applications*, pages 149–152, 2010. 107

[98] M. Radványi, B. Varga, and K. Karacs. **Advanced crosswalk detection for the Bionic Eyeglass**. In *Proceedings of the 12th International Workshop on Cellular Nanoscale Networks and Their Applications*, pages 1–5. IEEE, 2010. 107, 111

[99] B. Varga and K. Karacs. **Towards a Balanced Trade-off Between Speed and Accuracy in Unsupervised Data-driven Image Segmentation**. *Machine Vision and Applications*, 2012. Under review. 108

[100] K. Karacs, A. Lazar, R. Wagner, D. Bálya, T. Roska, and M. Szuhaj. **Bionic Eyeglass: An Audio Guide for Visually Impaired**. In *Proceedings of the 2006 IEEE Biomedical Circuits and Systems Conference*, pages 190–193. IEEE, 2006. 111

[101] S. Tőkes, V. Szabó, L. Orzó, P. Divós, and Z. Krivosija. **Digital Holographic Microscopy and CNN Based Image Processing for Biohazard Detection**. In *Proceedings of the 11th International Workshop on Cellular Neural Networks and Their Applications*, pages 8–8. IEEE, 2008. 111

[102] B. Varga. *Color-based Object Segmentation for Water Quality Surveillance*. Thesis for Informatics Specialist in Bionic Computing, Pázmány Péter Catholic University, 2011. 111

DOI:10.15774/PPKE.ITK.2013.005

Appendix A

Additional High Resolution Evaluation Examples for the Parallel System

This chapter contains two additional examples for the output of the parallel segmentation framework evaluated in Subsection 3.4.2. Figures A.1 and A.2 show the results of the segmentation and merging phases utilizing the two corresponding system settings on high resolution input images.

DOI:10.15774/PPKE.ITK.2013.005

A. ADDITIONAL HIGH RESOLUTION EVALUATION EXAMPLES FOR THE PARALLEL SYSTEM

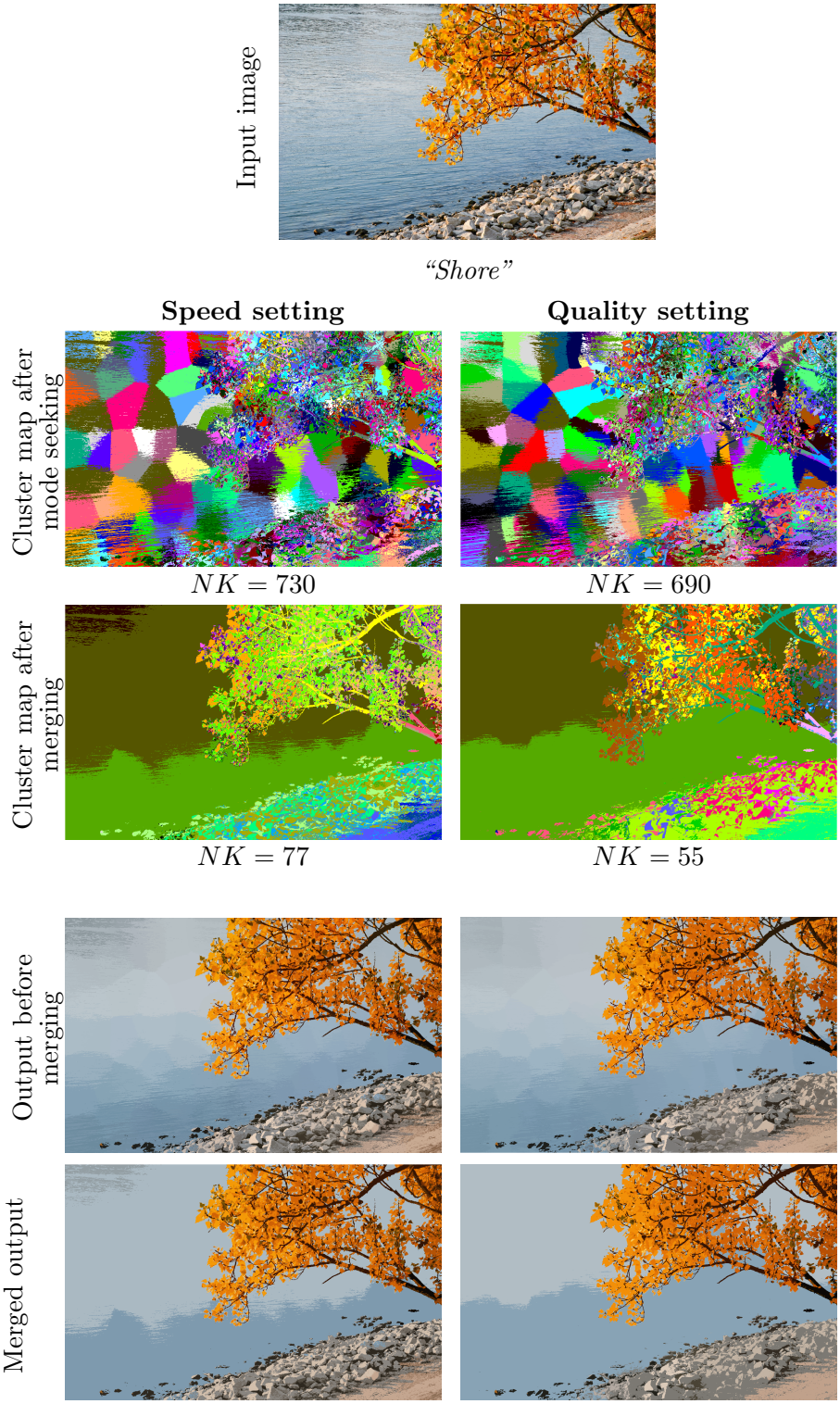


Figure A.1: A second high resolution segmentation example from the 15 image corpus used for the evaluation of the parallel framework. For the sake of better visibility, the extent of the clusters is also displayed in the form of cluster maps before and after the merging procedure. NK refers to the number of clusters.

DOI:10.15774/PPKE.ITK.2013.005

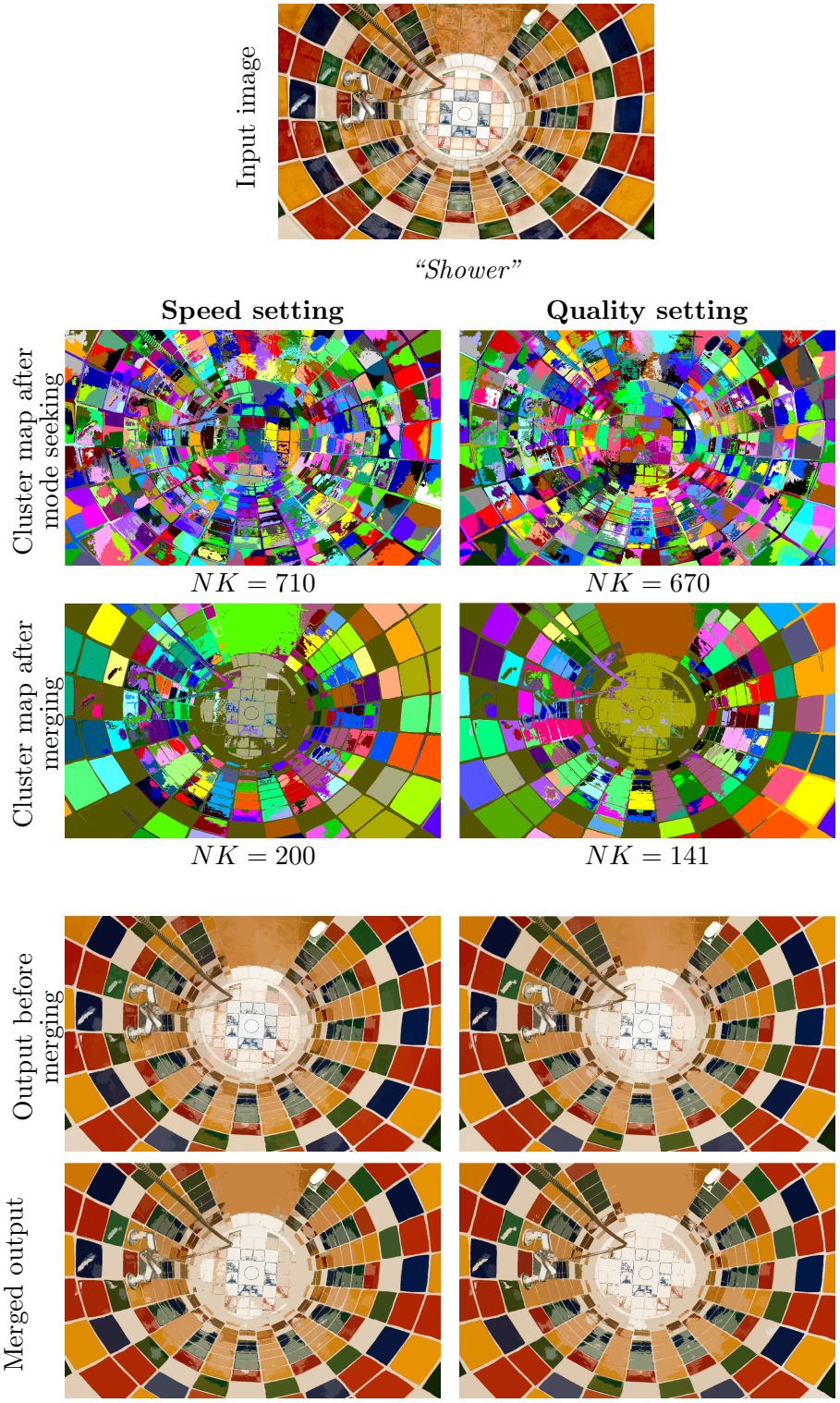


Figure A.2: A third high resolution segmentation example from the 15 image corpus used for the evaluation of the parallel framework. For the sake of better visibility, the extent of the clusters is also displayed in the form of cluster maps before and after the merging procedure. NK refers to the number of clusters.

DOI:10.15774/PPKE.ITK.2013.005

A. ADDITIONAL HIGH RESOLUTION EVALUATION EXAMPLES FOR THE PARALLEL SYSTEM

DOI:10.15774/PPKE.ITK.2013.005

Appendix B

Additional BSDS Evaluation Examples for the Adaptive System

This chapter contains three additional examples for the output of the adaptive segmentation framework evaluated in Subsection 4.5.1.

DOI:10.15774/PPKE.ITK.2013.005

B. ADDITIONAL BSDS EVALUATION EXAMPLES FOR THE ADAPTIVE SYSTEM

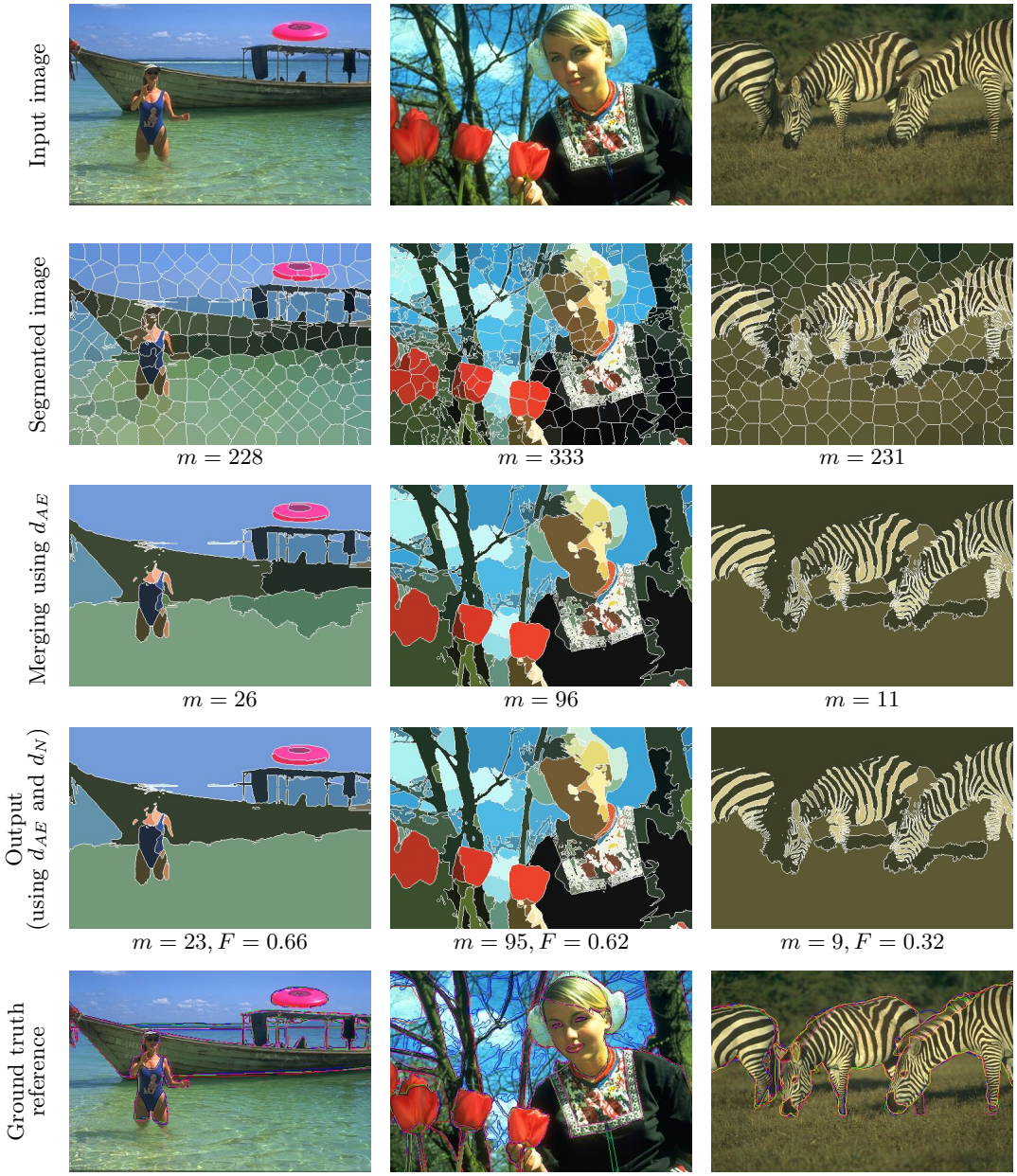


Figure B.1: Additional segmentation examples from the test set of the BSDS500. The first row contains the input images, rows 2 to 4 show the results obtained at the end of certain stages of the procedure. The number of clusters is denoted by m, the F-measure of the segmentation output is denoted by F. Row 5 shows the boundaries of multiple ground truth segmentations provided as reference, with different colors. Segmentation was done using the OPTIMAL parametrization.

DOI:10.15774/PPKE.ITK.2013.005

Appendix C

Additional High Resolution Evaluation Examples for the Adaptive System

This chapter contains twenty additional examples for the output of the adaptive segmentation framework evaluated in Subsection 4.5.2.

DOI:10.15774/PPKE.ITK.2013.005

C. ADDITIONAL HIGH RESOLUTION EVALUATION EXAMPLES FOR THE ADAPTIVE SYSTEM

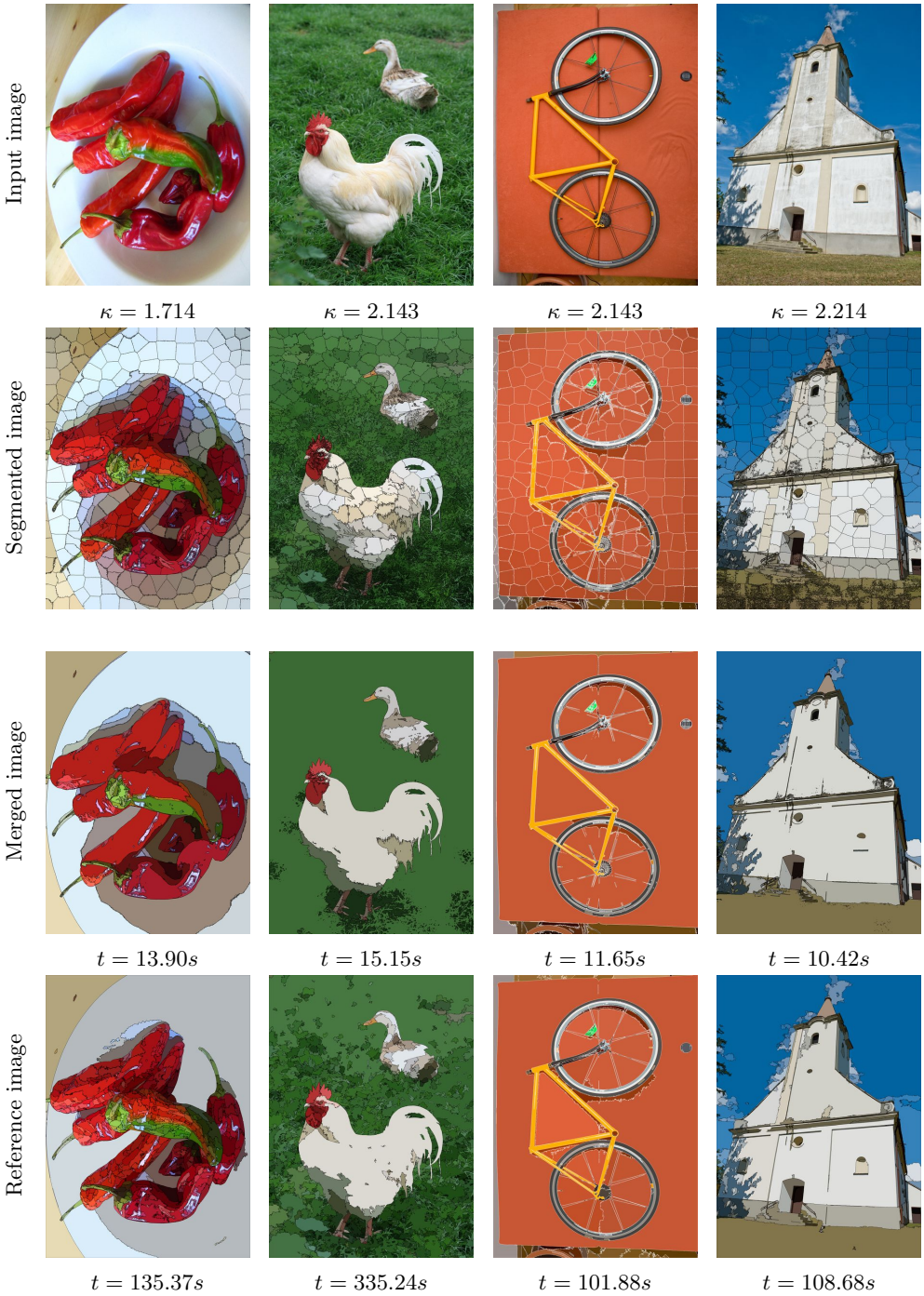


Figure C.1: Additional segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

Figure C.2: Additional segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

C. ADDITIONAL HIGH RESOLUTION EVALUATION EXAMPLES FOR THE ADAPTIVE SYSTEM

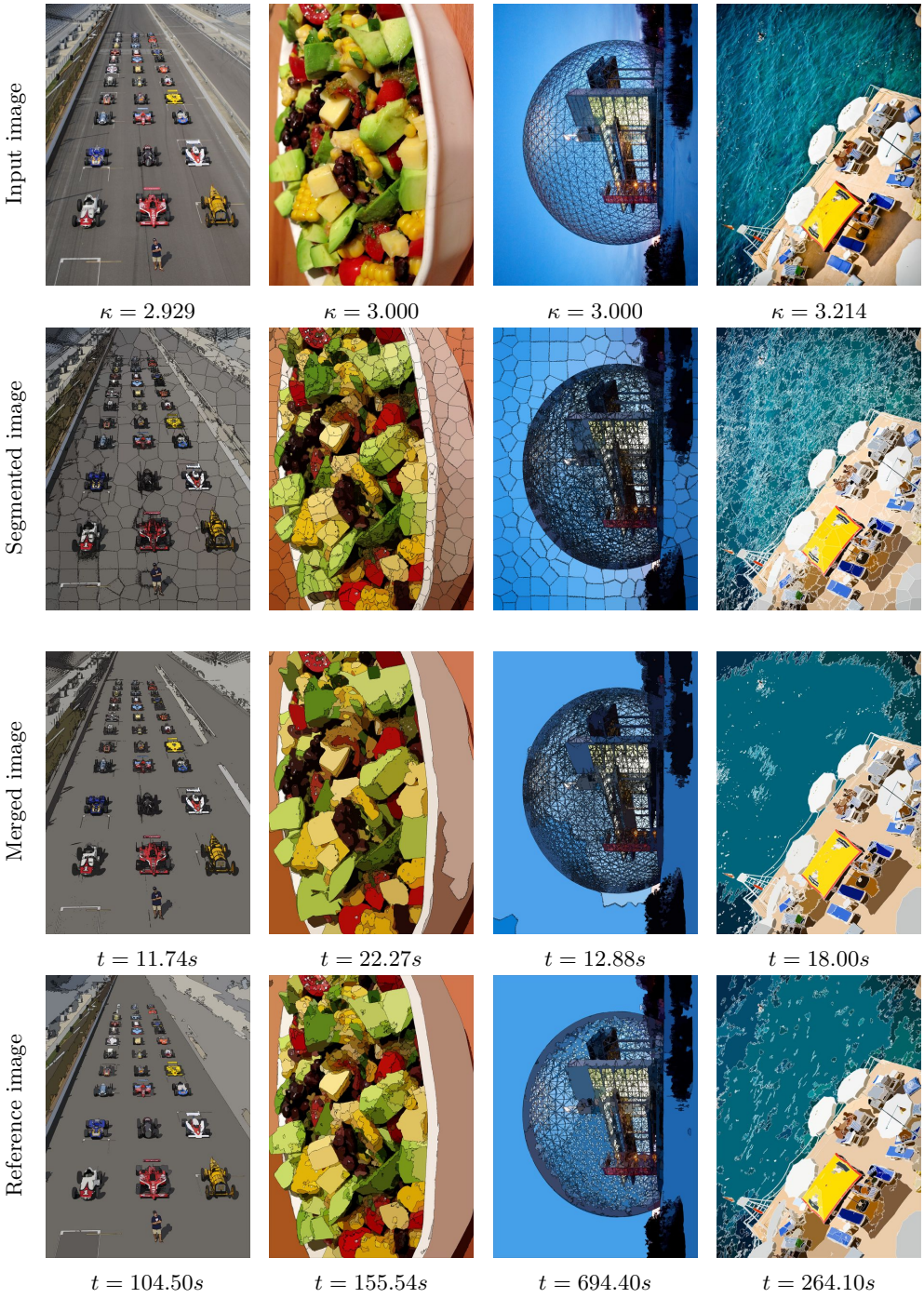


Figure C.3: Additional segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

Figure C.4: Additional segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.

DOI:10.15774/PPKE.ITK.2013.005

C. ADDITIONAL HIGH RESOLUTION EVALUATION EXAMPLES FOR THE ADAPTIVE SYSTEM

Figure C.5: Additional segmentation examples from the high resolution image set. Input images are found in row 1, row 2 shows the output of the segmentation phase using the HD-OPTIMAL parametrization. Merged images can be seen in row 3, finally, row 4 contains the segmentation results of the reference system using the high setting. For the sake of visibility, cluster boundaries marked with black or white (depending on which is more salient). The kappa-index (κ) is indicated for each image, along with the running time (t) of the algorithms.