CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY IN PRAGUE

DOCTORAL THESIS

ISSN 1213-2365

Analysis of microscopy images

Automatic analysis of gene expressions in Drosophila microscopy images

Jiří Borovec

jiri.borovec@fel.cvut.cz

CTU–CMP–2017–07

December 2017

Available at
ftp://cmp.felk.cvut.cz/pub/cmp/articles/borovec/Thesis-TR-2017-07.pdf

Thesis Advisor: Prof. Jan Kybic

This Doctoral Thesis presented to the Faculty of the Electrical Engineering of the CTU in Prague in fulllment of the requirements for the Ph.D. Degree in Study Programme No. P2612 Electrical Engineering and Information Technology, branch No. 3902V035 - Articial Intelligence and Biocybernetics.

Research Reports of CMP, CTU in Prague, No. 7, 2017

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Analysis of microscopy images

Automatic analysis of gene expressions in Drosophila microscopy images

Jiří Borovec

December 2017

Abstract

This work aims at providing a fully automated processing and analysis pipeline for microscope images of Drosophila melanogaster (fruit fly), which is a highly valuable study subject for biologists because of high gene similarity with mammals and short life cycle. In particular, the biologists are interested, for instance, in observing gene expressions in the early stages of fly development and later in the larval stage. The used processing pipeline for Drosophila ovaries consists of (i) segmenting individual eggs in a chain; (ii) selecting eggs of particular developmental stages and (iii) aligning them with a characteristic prototype; (iv) estimating an atlas of observed gene location; and (v) performing data mining (statistical observation) for different genes. In this work, we address steps (i) and (iv) of this procedure to improve the poor experimental performance of images segmentation and decomposition methods.

Proper ovary segmentation is a challenging task for several reasons: high similarity between tissue classes, highly structured egg chamber with individual eggs frequently touching. We decompose egg segmentation step into three subtasks: (a) semantic image segmentation into classes with biological meaning, (b) object center detection for each possible egg, and (c) instance segmentation of individual eggs initialized by estimated centers on pre-segmented images. First, we use image segmentation on superpixels (instead of pixels) with Graph Cut regularization for higher compactness. Computing features on superpixels increase representation expressiveness and speed-up further optimization. We introduce an object center detection based on label histogram and object shape description with rotationally invariant ray features. On top of the center detection, we formulate and use an ellipse fitting to approximately estimate object boundaries (ovary chambers) to select one of the center candidates for each object. Later, we extended the region growing segmentation by using superpixels together with a learned shape prior. Finally, we address the sensitivity of the standard decomposition methods for atlas (a small set of non-overlapping spatial patterns) estimation to noise, by proposing a novel Binary Pattern Dictionary Learning, which benefits from using spatial information. Then each input images can be represented by a union of these patterns using Dice measure.

In comparison to standard techniques for the particular tasks, our proposed methods increase performances - accuracy and time, and help to automate the utilized processing pipeline as we present experimental results on both real datasets of Drosophila ovary and imaginal discs. All proposed methods were developed with a specific application in mind; however, they can be easily generalized and applied to other domains not only in biomedical or microscopy, such as satellite imaginary, detection objects (e.g. cars) in a scene or signal compression.

Keywords:

Superpixels, semantic image segmentation, instance image segmentation, (un)supervised learning, object center, ellipse fitting, region growing, ray features, label histogram, clustering, shape prior, pattern extraction, atlas, decomposition, Graph Cut, microscopy imaging, Drosophila, ovary, egg chamber, imaginal disc, gene locations.

Abstract

Analýza mikroskopických snímků je jedním z klíčových nástrojů biologů při pozorování genových expresí v průběhu raných stádií životního cyklu mouchy octomilky (Drosophila melanogaster), v počáteční fázi (vajíčka) a později v larvální fázi (zkoumány jsou části zvané imaginální disky). Námi použitá sekvence metod pro tuto analýzu se skládá z následujících kroků: (i) identifikace a segmentace jednotlivých objektů (ii) určení vývojových stupňů (tříd) pro každé vajíčko; (iii) prostorové zarovnání objektů dané třídy na společný vzor; (iv) vytvoření atlasu genových aktivací na základě pozorování; a (v) vytěžování dat a statistické vyhodnocení pro různé geny. V této práci se blíže věnujeme krokům (i) a (iv), kde používané standardní metody nedosahují dostatečně dobrých výsledků, a proto pro potřeby této automatické analýzy navrhujeme nové metody pro segmentaci a rozklad obrazu na omezený počet prostorových vzorů.

V segmentaci vajíček je hlavní výzvou vysoká podobnost mezi jednotlivými druhy buněčných tkání a častý vzájemný dotyk spolu s deformací sousedících vajíček. A proto jsme krok (i) rozložili do tří navazujících dílčích úkolů: (a) Obraz je segmentován do několika tříd podle druhu tkáně na základě lokální informace. Segmentace je prováděna na úrovni superpixelů (namísto pixelů) s využitím Graph Cut optimalizace pro vyšší kompaktnost. Dále, příznaky počítané na superpixelech zvyšují expresivitu a zrychlují následující optimalizaci. (b) Na segmentovaných snímcích jsou detekovány středy jednotlivých objektů (vajíček). Jako příznaky jsou použity histogram tříd a popis tvaru potencionálního objektu pomocí rotačně invariantních paprskových deskriptorů (ray features). Po detekci středů následuje aproximace jednotlivých vajíček elipsou (proložením okrajových bodů elipsou), která slouží k určení vícenásobných detekčních hypotéz pro jedno vajíčko a přibližné segmentaci. (c) Následně jsou nalezeny obrysy i jednotlivých vajíček pomocí vylepšené metody region growing z odhadnutých středů z úkolu (b) a na základě segmentace z úkolu (a). Tato metoda je také definována na úrovni superpixelů a používá naučený tvar předpokládaných objektů. V kroku (iv), pro vytváření atlasu genových aktivací (vzorů) je navrhnuta nová metoda - Binary Pattern Dictionary Learning, která využívá prostorového uspořádání v obraze, a tím eliminujeme nevhodnou citlivost na šum u stávajících metod. Pomocí takovéhoto atlasu je možné každý segmentovaný obraz reprezentovat s pomocí jen krátkého vektoru vyznačujícího aktivní vzory.

V porovnání se standardními metodami pro jednotlivé dílčí úkoly (i-a,i-b,i-c) a krok (iv) námi navržené metody dosahují lepších výsledků z pohledu přesnosti i časové náročnosti, což je další krok směrem ke zcela automatické analýze mikroskopických snímků octomilky. Výsledky navržených a standardních metod ukazujeme a porovnáváme na reálných mikroskopických snímcích vajíček a imaginálních disků octomilky. Všechny navržené metody byly vyvinuty s ohledem na konkrétní využití, lze je však snadno upravit a použít i v jiných oblastech, nejen v biomedicíně a mikroskopii, ale například i pro zpracování satelitních obrazů.

Klíčová slova:

Superpixely, segmentace obrazu, učení s učitelem, středy objektů, elipsa, růst oblasti, paprsky, histogram tříd, shlukování, tvarování, extrakce vzorů, atlas, rozklad, Graph Cut, mikroskopické obrázky, octomilka, vajíčka, imaginální disk, umístění genů.

Acknowledgements

First of all, I would like to express my gratitude to my dissertation thesis supervisor, prof. Jan Kybic. He gave me a chance to work in such exciting field — medical imaging, even with my zero experience on the beginning. I thank for his insights during my research and help me with numerous problems and professional advancements.

I would like to thank very much to prof. Vaclav Hlavac who accepted me in his excellent group — Center for Machine Perception (CMP) and was a great inspiration to me with his networking and social insight. Also, great thanks go to prof. Akihiro Sugimoto who accepted me for an internship in National Institute of Informatics (NII) in Tokyo, gave me many valuable pieces of advice and showed me new perspectives.

Special thanks go to the staff of the Department, who maintained a pleasant and flexible environment for my research. I would like to express thanks to the department and group managers for providing a funding for my research. My research has also been partially supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS12/190/OHK3/3T/13 and SGS15/154/OHK3/2T/13, by the Czech Science Foundation as project No. P202/11/0111 and 14-21421S. I also would like to acknowledge the support of the NII internship program, which helped me finish my Ph.D. studies. The images presented in this thesis were provided by Pavel Tomancak's group, MPI-CBG, Germany.

I would like to express thanks to my CMP colleagues, namely Dr. Jan Svihlik, Dr. Boris Flach, Dr. Michal Reinstein, Dr. Karel Zimmermann, Michal Busta, Martin Dolejsi and many others, for their valuable support, comments, and ideas. I would also like to thank Dr. Jan Hering and Dr. Boris Flach for proofreading the text of the thesis. Thanks also go to prof. Arrate Munoz-Barrutia and prof. Carlos Ortiz-de Solorzano who help me with my first academic paper during my internship in CIMA.

Finally, my most enormous thanks go to my family, for their infinite patience with my long study and support.

Contents

Abbreviations

US	Ultrasound (Sonography)
MRI	Magnetic Resonance Imaging
SLIC	Simple Linear Iterative Clustering
LUT	Look up Table
CNN	Convolutional Neuronal Networks
MRF	Markov Random Field
GMM	Gaussian Mixture Model
EM	Expectation-Maximization
RF	Random Forest
KNN	k-Nearest Neighbors
SVM	Support Vector Machine
GC	Graph Cut
DBSCAN	Density-Based Spatial Clustering
RANSAC	Random sample consensus
RG	Region Growing
ARS	Adjusted Rand Score
PCA	Principal Component Analysis
ICA	Independent Component Analysis
DL	Dictionary Learning
NMF	Non-negative Matrix Factorization
BPDL	Binary Pattern Dictionary Learning

Symbols

Image related

Ω	set of pixels (image plane)
I	input image function $I : \Omega \to \mathbb{R}^m$
$\mathbb{L}$	set of labels

Superpixels

η	original regularization
ξ	proposed regularization
ν	initial superpixel size
s	superpixel
S	set of superpixels $s \in S$
D	weighted distance
$d_{\{c,s\}}$	color and spatial distance respectively
Ω_s	pixels belonging to a superpixel s, $\Omega_s \subset \Omega$

Segmentation & Region growing

$\mathbf{y}_\Omega$	pixel-wise segmentation function $\mathbf{y}_\Omega : \Omega \to \mathbb{L}$
Y_Ω	orderly set of pixel-wise segmentation $Y_\Omega = \mathbf{y}_\Omega(\Omega)$
$\mathbf{y}$	superpixel segmentation function $\mathbf{y} : S \to \mathbb{L}$ with abbrev. for $\mathbf{y}_s = \mathbf{y}(s)$
Y	orderly set of superpixel segmentation $Y = \mathbf{y}(S)$
x	feature vector
X	set of features $x_s \in X$ for all superpixels
$d_{\{M,E,T\}}$	Manhattan, Euclidean and Tchebychev distance respectively
U, B	unary (data) and binary (pairwise) term for GC respectively
g	image descriptor
r	vector of ray distances
c	vector of object centers
m	statistical shape model $m = [c, r, \Theta, w]$
M	mixture of shape model
w	vector with model weights
q	cumulative probability of spatial prior

Dictionary Learning

g	image appearance (binary association to a class)
$\mathbf{G}$	set of images $g \in \mathbf{G}$
$\mathbf{y}_\Omega$, Y_Ω	atlas (binary patterns, segmentation)
$\mathbf{w}$	vector with binary weights
$\mathbf{W}$	matrix with binary weights $\mathbf{w} \in \mathbf{W}$

“Success is no accident. It is hard work, perseverance, learning, studying, sacrifice and most of all, love of what you are doing or learning to do."

- Pelé

“If you can’t fly then run, if you can’t run then walk, if you can’t walk then crawl, but whatever you do you have to keep moving forward."

- Martin Luther King, Jr.

虎穴に入らずんば虎子を得ず。

- Japanese idiom

1 Introduction

The image analysis is a critical phase in many medical and biological applications and treatments. This thesis is focusing on medical imaging, namely in microscopy images. Compared to other applications the main characteristic of medical imaging is an appearance of noise, image distortion and a small number of annotated data. On the other hand, the importance of using some image processing methods and tools is increasing with the rapid growth of image resolutions captured with new technologies and amount of sensed images compare to stagnating number of human experts performing the analyses manually. It shows the high demand for (semi)automatic image analyses which would solve some tasks entirely independently or at least significantly speed up the process and assist to a human expert.

In this thesis, we aim at automatic image analysis of microscopy images of Drosophila, which is an important study subject in genetic biology. The used processing pipeline starts with sensed images and ends by extracting an atlas of gene activations. The methods presented in this work are image segmentation on superpixels, estimation object centers, region growing on superpixels and final decomposition of a stack of segmented images into a small number or non-overlapping patterns. Note that similar pipeline can be used in other research and notably proposed methods can also be applied in other fields.

In this chapter, we explain the need for new processing methods and formulate the related problems we face in this work. In the end, we summarize the main contributions of this work and present the structure of the thesis by referring to my publications.

1.1 Drosophila

For nearly a century common fruit fly — *Drosophila melanogaster*, has played a significant role in genetics because it constitutes a suitable model for studying many aspects of ova de-

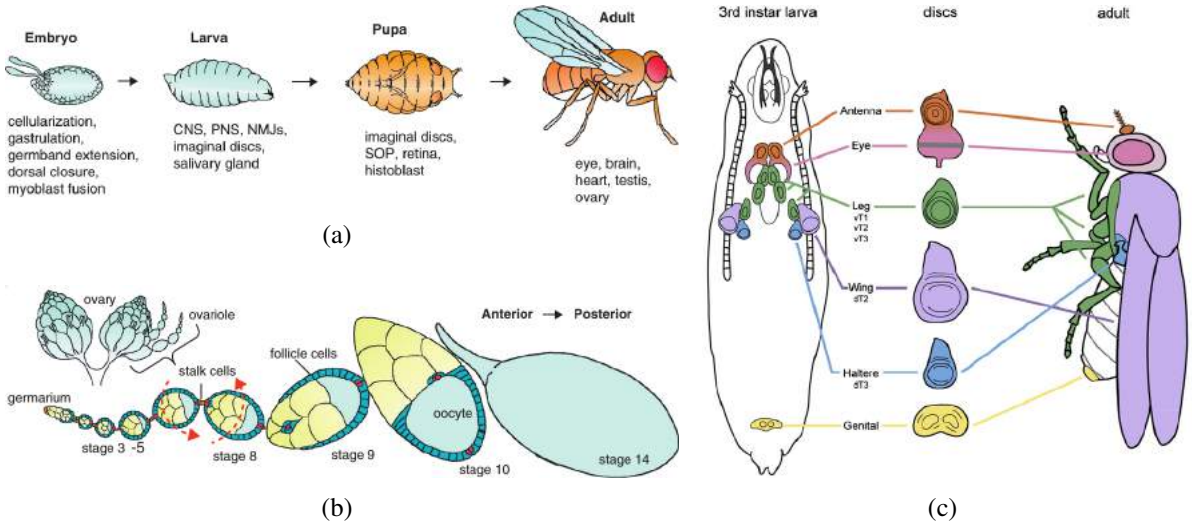


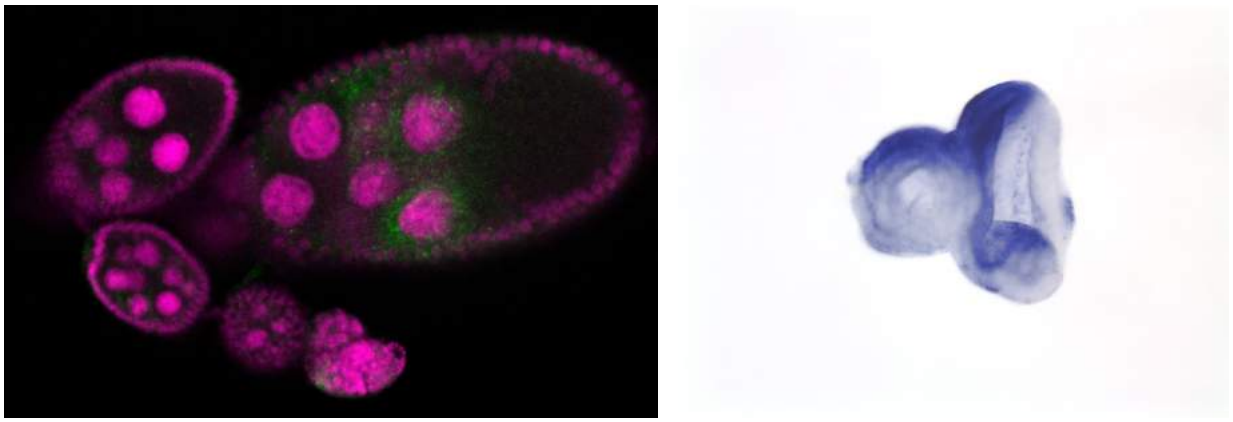
Figure 1.1: *(a)* Schema [7] of the complete Drosophila development from ovary to adult. *(b)* Development stages of Drosophila ovary. [7] *(c)* Visualization [8] of Imaginal disc location and shape in larva phase and transition to section in adult.

velopment such as birth and destruction of germ cells, the role of stem cells, or knowing the functioning of messenger ribonucleic acid (mRNA) [1].

Some studies have shown Drosophila shares many fundamental properties with mammals [2]. It is estimated that 700 out of 2309 genes that are associated with human diseases have a homolog in Drosophila genes [3]. Therefore, the understanding of its cellular development may allow scientists to extend experiences to more complex organisms. For instance, Reitel et al. [4] found out that some protein sequences in mammals are related to proteins in Drosophila. Potter et al. [5] suggested that Drosophila may be useful in the study of tumorigenesis due to similarities with cellular processes in mammals, whereas Rudrapatna et al. [6] concluded the investigation of tumor suppressors in Drosophila might lead to a better understanding of fundamental processes of cancer in human beings.

The matured egg chambers allow biologists to track a particular egg from a single stem cell to an ovum, see Figure 1.1(a). A female Drosophila has a pair of ovaries, each consists of roughly 18 independent ovarioles that are similar to a chain-like structure (see Figure 1.1(b)), and every link represents a single egg chamber [9]. The anterior tip of each ovariole is known as a germarium, which is a structure composed of stem cells. The developing egg chambers are connected by somatic stalk cells and follow the germarium with the most mature egg at the most distal point [10]. Typically, an egg chamber contains sixteen germ cells; one of these germ cells develops as an oocyte, and the rest cells become nurse cells. The set of cells is surrounded by follicle cells [11].

In later stage of Drosophila development [12], an imaginal disc is one of the parts of a holometabolous insect larva that will become a portion of the adult insect during the pupal transformation (see Figure 1.1(a)). [7, 13] Contained within the body of the larva, there are pairs of



(a) (b)

Figure 1.2: Sample microscopy images with stained gene activation. *(a)* Chain of Drosophila ovaries with gene activation highlighted in the green channel. *(b)* Drosophila imaginal disc with gene activation highlighted by dark blue color.

discs that will form sections of adult fly, such as the eye, legs, wings or antennae or other structures, see Figure 1.1(c). The role of the imaginal disc in insect development was investigated — observing gene expression from microscopy images stain by RNA in situ hybridization [14]. Each gene was imaged a number of times, with the aim of determining the role of the gene in the development process [15].

1.2 Motivation

Recent advances in microscopy imaging have created new opportunities to study complex genetic interactions in Drosophila [16], for instance, discovering gene patterns in various development stages. Each gene or group of genes can be highlighted by a special stain to be easily tracked in microscopy images, see Figure 1.2. There is an assumption that individual genes are active in particular development stages and also in specific locations of the tissue (for instance ovary or imaginal disk). However, there are hundreds of genes to be analyzed, and from the roots of this task, we assume observation each gene several times because the gene expression (activation) is partially a stochastic event [17].

Many studies of large scale mapping of the gene expressions have been performed in embryos [17–29] as well as in imaginal discs [15, 30–35], which are essential for the initial development of the adult fly. Such studies aim to discover the functionality of specific genes, which is of paramount importance in basic biological research with possible therapeutic applications in medicine. Usually, the image analyses (e.g., segmentation) is performed manually, or with some semi-automatic tools, so it is a laborious and time-consuming task which is limiting for processing more significant amount of images and obtain stronger results.

The motivation of this work is to provide a set of tools for automatic image analysis of spatial

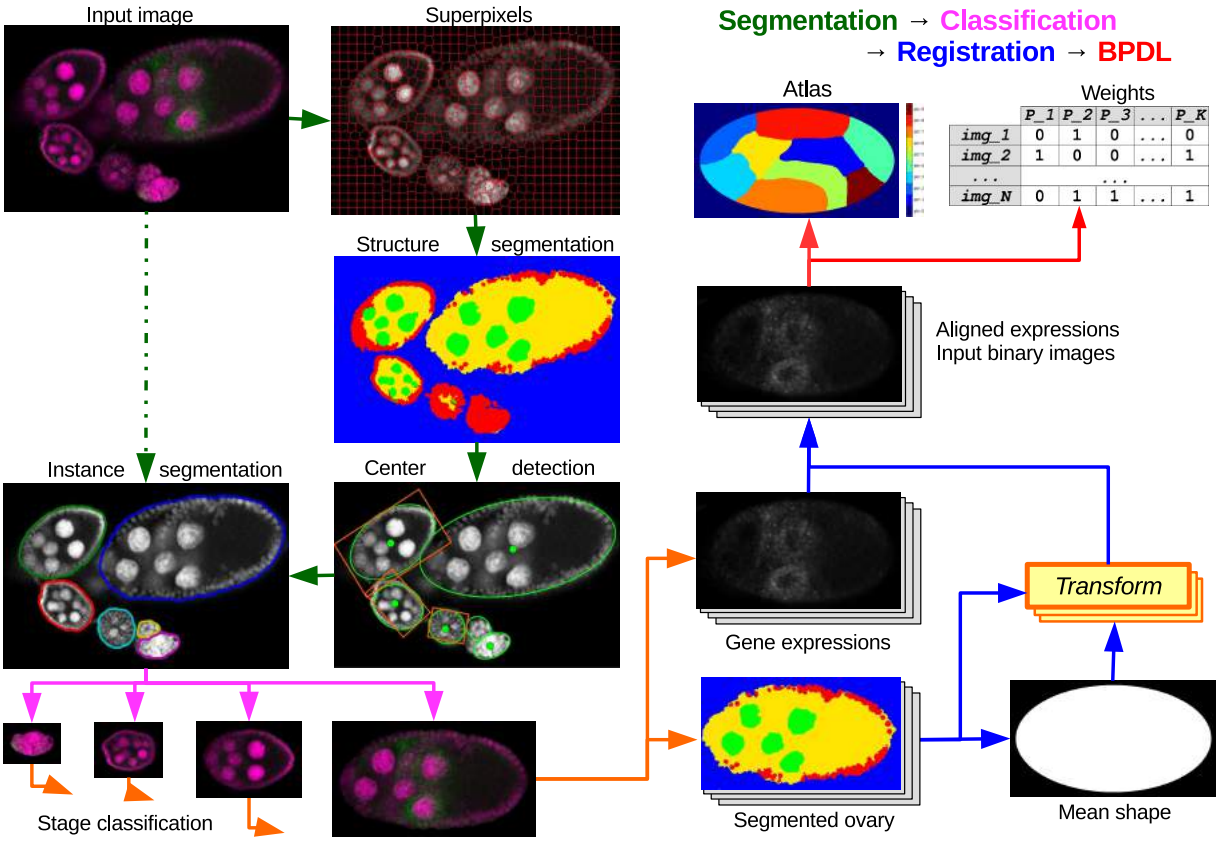


Figure 1.3: A flowchart of the complete pipeline for processing images of Drosophila ovary: (i) supervised superpixel segmentation, egg center detection, and region growing; (ii) classification ovary chamber by stages and group them; (iii) registration of binary segmented eggs onto a reference shape (chamber prototype); (iv) atlas estimation from aligned gene expressions.

and temporal patterns of gene expression during the egg-chamber development (oogenesis) and imaginal discs of the common fruit fly. It is a common practice to collect thousands of such images to analyze different genes of interest [17, 22, 36, 37]. Once the individual egg chambers are localized and segmented [38–40], they are grouped by their developmental stage [41], aligned together to a common prototype [42, 43], and the gene activations are analyzed [44].

1.2.1 Image segmentation

Image segmentation is an image analysis method which decomposes input images into a small number of meaningful regions which usually share similar property. This segmentation is also commonly called 'semantic segmentation', see Figure 2.1(b). Image segmentation is a key image processing task with a vast range of applications, namely in biomedical imaging [45–47], including organ segmentation [48], object detection and extraction [17], or preprocessing for feature extraction or object tracking. The segmentation of medical images has faced some specifics, additional specifications and challenges compared to the typical applications in Computer Vi-

sion. The particular challenges are an appearance of noise, distortions and deformations, and also absence or a large set of annotated images which are beneficial for supervised learning. This task is popular in computer vision and deep learning, and it was addressed by several methods based on Convolutional Neuronal networks (CNN) [49, 50].

Also, with the steadily growing number and spatial resolution of the images to be processed, we need methods which are simultaneously fast and accurate. The size of microscopy images may be up to 10^{10} pixels for a 2D a histology slice or 3D fluorescent microscopy image of an ovary. Current segmentation techniques can be very slow, especially if their computational complexity is not linear concerning the number of pixels, as is the case, e.g., for many MRF optimization techniques. One way to address this issue is to work on superpixels [51] instead of pixels which we review later in the following Chapter 2, in particular, Section 2.1 and 2.2. We also show how superpixels may provide increased robustness to noise.

1.2.2 Individual object segmentation

Here, we address the first step of the image processing pipeline - segmenting individual object in the input image and marking the rest as background. This image segmentation is commonly called 'instance segmentation', see Figure 2.1(d). The egg segmentation is a crucial step in Drosophila ovary analysis such as eggs stage classification which assumes only single egg in an image and further creating atlases of locations of possible gene activations [17], see Section 1.2.3.

This task became very popular in computer vision, and several methods based on Convolutional Neuronal networks (CNN) have been recently proposed [52–54]. Unfortunately, all these deep approaches require a large amount of annotated training data which is always an issue in medical imaging where the annotated images are counted in tens not in thousands or millions as it is common in other applications [55].

This problem has several challenging aspects. First, the objects (eggs) are highly structured and cannot be easily distinguished by texture or intensity due to high similarity between pairs of tissue classes (background—cytoplasm and nurse—follicular cells). Second, there are several eggs in the image, often touching, with unclear boundaries between them. Third, the algorithm should be fast, as there is a high number of images to be processed. Note also that identifying individual eggs with mutually similar appearance is more challenging than a standard binary or multi-class segmentation [39, 56].

The most straightforward way of segmenting Drosophila eggs would be to apply region growing from a manually or automatically marked seed point in the approximate center of each egg until it reaches the boundary or touches a neighboring egg. However, the real boundary cannot be quickly recovered because of the various structures inside each egg. Template matching methods would suffer from the different appearance of the eggs' inside, for example, the high number of various positions of the nurse cells as illustrated on Figure 1.3. Also from 2D perspective then a number of nurse cells can vary. A texture-based structure segmentation method for egg chamber images was described in [56] but there is no simple way how to obtain a segmentation of individual eggs directly. Unfortunately, the semantic segmentation is not sufficient for identifying individual egg chambers, for example, using watershed segmentation [57]. Another standard approach is to post-process the foreground/background segmentation using mathematical mor-

phology and connected component analysis, but in this case, it turned out not to be sufficiently robust due to the touching objects (see Section 7.3). Nevertheless, such tissue segmentation can be used as the initial step of our approach.

1.2.3 Aggregation of gene expressions

The final step of the considered image analyses pipeline are to understand the role of the different genes by comparing locations, where the genes are expressed, with the known information about the function of the various areas. To reduce the dimensionality of the problem and to enable an efficient statistical analysis, the observed spatial expressions are described by a set of labels from a limited, application specific dictionary — an 'atlas'. Example labels for the leg imaginal disc are 'dorsal' or 'ventral' but also 'stripes' or 'ubiquitous'. Given such sets of labels, correlations with gene ontologies can be then found using data mining methods [58–66] which are beyond the scope of this work.

The dictionary, as well as the labels for individual images, are usually determined manually or semi-manually [17, 22, 67], which is extremely intensive work since it requires two passes through the set of images, first to establish the dictionary of spatial locations and then to perform the labeling of each image using the dictionary. Note that there are several thousands or hundreds of thousands of images to be processed.

This task can be addressed as (linear) decomposition problem. Unfortunately, the standard approaches suffer from an appearance of noise even in segmented images and deformations of the observed pattern in individual images. Furthermore, even aligning all images to a common prototype does not prevent displacements inside such. Moreover, any method introducing any spatial compactness to overcome these issues become resources and time highly demanding.

1.3 Problem Statement

The used image analyses pipeline consist of several subtasks: instance image segmentation of individual objects (eggs), stage classification of particular development stages (ovaries) or tissue type (imaginal discs), image registration of a stack of images into a common prototype and atlas extraction of binary patterns of gene expressions. In this work, we deal with two tasks from the list - instance segmentation and atlas extraction. The other tasks are out of the scope of this works since they can be sufficiently solved by other methods, such as stage classigication [41] or image registration [42, 43].

The instance segmentation — the input is an image with multiple object instances (e.g., Drosophila ovaries), and the output is an image segmentation of a small number of regions where each object is assigned a unique label. As described earlier, we split this task into three subtasks: (i) semantic segmentation - segmenting an input image into a small number of classes with a biological meaning (see Figure 2.1(b)); (ii) detecting individual object centers in the pre-segmented image (semantic segmentation) - a point is assigned to each object (each object should be marked by only one point), and it is located in approximately object's center (see Figure 2.1(b)); and (iii) segmenting individual object instances (e.g., Drosophila egg chambers) in the pre-segmented

(semantic segmentation) image initialized by approximate object centers - region growing (see Figure 2.1(d)).

The second task is aggregation gene activations — location patterns in both ovaries and imaginal disc. Here the input is a set of binary segmented gene activation where all images are aligned to a common (template) prototype and to be of the same (biological) kind — development stage and tissue class. The output for each set of images is an 'atlas' of binary patterns (non-overlapping regions) — segmentation and (sparse) binary encoding for each image marking presented patterns from an estimated pattern dictionary.

1.4 Contribution of the Dissertation Thesis

The main contributions of this thesis towards extending state of the art in scene superpixel segmentation, region growing, and atlas extraction are:

- **Enhancement of superpixels computing** in superpixels representativeness and processing time (see Figure 4.3).
- **Supervised and unsupervised superpixels segmentation with Graph Cut regularization** used for semantic segmentation, it uses more reliable features and speeds-up learning and optimization together with enforcing segmentation compactness (see Figure 5.8 and 5.9).
- **Object center detection and ellipse fitting**, it uses efficient center point description (label histogram and rotation invariant ray features) and ellipse fitting to object boundaries eliminates multiple center proposal inside a single object (see Figure 6.6).
- **Region growing on superpixels with a learned shape prior** used for instance segmentation from previously detected centers, it introduces faster region growing into possible shapes (see Figure 7.10).
- **Binary pattern dictionary learning algorithm** extracts compact non-overlapping patterns (atlas) in a large stack of aligned binary images and represent observations in each image via sparse binary encoding (see Figure 8.4).

In addition, we release the source code of the presented methods together with the datasets used for training and validation.

Source code - publicly available implementation for all previous methods.

- jSLIC[1] - a plugin for superpixels in ImageJ.
- ImSegm[2] covers following methods: (Un)Supervised segmentation, Object center detection and Region Growing.

[1] http://imagej.net/CMP-BIA_tools
[2] http://borda.github.io/pyImSegm

- BPDL[3] - a package for Binary Pattern Dictionary Learning.

Datasets - annotated images for both Drosophila datasets (ovary and imaginal disc) from two perspectives as semantic (see Figure 2.1(b)) and instance (see Figure 2.1(d)) segmentation.

- Ovaries[4] with semantic and instance annotations.
- Imaginal disc[5] with semantic annotation.

For more details about datasets see Chapter 3.

1.5 Structure of the Dissertation Thesis

The thesis is organized into chapters as follows:

1. *Introduction* describes the motivation behind our efforts together with our goals. There is also a list of contributions of this thesis.
2. *State-of-the-Art* introduces the reader to the necessary theoretical background and surveys the current state-of-the-art for each particular image processing task.
3. *Materials* describes used microscopy images together with user annotation for ovaries and imaginal disc as well as annotation for particular image processing tasks.
4. *Superpixel extraction [68]*: We present the particular improvements on the superpixel clustering algorithm—SLIC (Simple Linear Iterative Clustering). The main contribution of the jSLIC (our implementation as a plugin in ImageJ) is a significant speed-up of the original clustering method, transforming the compactness parameter such that the value is image independent, and a new post-processing step (after clustering) which now gives more reliable superpixels—the newly established segments are more homogeneous.
    - Jiri Borovec, Jan Kybic **jSLIC : superpixels in ImageJ.** In: *Computer Vision Winter Workshop*. Praha, Czechia. 2014.
      Authorship: 90—10
5. *Supervised and unsupervised superpixel segmentation [40]*: We present a fast and general multi-class image segmentation method on superpixels. We apply this segmentation pipeline to real-world medical imaging applications and present obtained performance. We show that unsupervised segmentation provides similar results to the supervised method, but does not require manually annotated training data, which is often expensive to obtain.

[3] http://gitlab.fel.cvut.cz/biomedical-imaging-algorithms/APDL
[4] https://drive.google.com/drive/folders/0BzDuwecaWm4KaXZhcVRjZmZ6LXc
[5] https://drive.google.com/drive/folders/1r4xe_fCXO99P51uItxVOwvKIFZmGai6X

- Jiri Borovec, Jan Kybic **Fully automatic segmentation of stained histological cuts.** In: *17th International Student Conference on Electrical Engineering*. Praha, Czechia. 2013.
  Authorship: 90—10
  The paper received the **Poster award**.
- Jiri Borovec, Jan Svihlik, Jan Kybic, David Habart; **Supervised and unsupervised segmentation using superpixels, model estimation, and graph cut.** In: *Journal of Electronic Imaging* 26(6). 2017. DOI: 10.1117/1.JEI.26.6.061610
  Authorship: 70—14—14—2; Impact factor (2012): 1.06

6. *Object center detection and ellipse fitting [38]*: An detection of individual egg chambers is reqiured for studying Drosophila oogenesis (egg chamber development), which is usually performed manually and so it is a time-consuming task. We present an image processing pipeline to detect and localize Drosophila egg chambers. We propose novel features for efficient center point description: label histogram computed from inter-circles regions describing the spatial relations and rotation invariant ray features approximating the object shape from a given point. We introduce ellipse fitting to object boundaries as maximal likelihood estimate which is utilized for elimination multiple center proposals inside single object/egg. Our proposal can detect most of the human-expert annotated egg chambers at relevant developmental stages with less than 1% false-positive rate, which is adequate for the further analysis.

   - Jiri Borovec, Jan Kybic, Rodrigo Nava **Detection and Localization of Drosophila Egg Chambers in Microscopy Images.** In: *8th International Workshop on Machine Learning in Medical Imaging (with MICCAI 2017)*. Cham, Switzerland. 2017.
     Authorship: 70—28—2

7. *Region Growing with Shape prior [39]*: Region growing is a classical image segmentation method based on hierarchical region aggregation using local similarity rules. Our proposed method differs from standard region growing in three important aspects: (i) it works on the level of superpixels instead of pixels, which leads to a substantial speedup; (ii) our method uses learned statistical shape properties that encourage plausible shapes; (iii) our method can segment multiple objects and ensure that the segmentation does not overlap. The problem is represented as an energy minimization and is solved either greedily, or iteratively using Graph Cuts. We evaluate performance on real microscopy images — Drosophila ovary.

   - Jiri Borovec, Jan Kybic, Akihiro Sugimoto. **Region growing using superpixels with learned shape prior.** In: *Journal of Electronic Imaging* 26(6). 2017. DOI: 10.1117/1.JEI.26.6.061611
     Authorship: 70—25—5; Impact factor (2012): 1.06

8. *Binary pattern dictionary learning [44]*: We present a binary pattern extraction method which accepts a large number of images, containing spatial expression information for

thousands of genes in Drosophila imaginal discs. We assume that the gene activations are binary and can be expressed as a union of a small set of non-overlapping spatial patterns, yielding a compact representation of the spatial activation of each gene. We compare proposed method to existing alternative methods on synthetic data and also show results of the algorithm on real microscopy images of the Drosophila imaginal discs.

- Jiri Borovec, Jan Kybic **Binary pattern dictionary learning for gene expression representation in Drosophila imaginal discs.** In: *Mathematical and Computational Methods in Biomedical Imaging and Image Analysis workshop at (with ACCV 2017)*. Cham, Switzerland. 2016.
  Authorship: 70—30
  The paper received the **Best paper award**.

9. *Conclusions* summarizes the results of our research, suggests possible topics for further research and concludes the thesis.

Authorship

I hereby certify that the results presented in this thesis were achieved during my own research, in cooperation with my thesis advisor Prof. Jan Kybic. The work presented in Chapter 7 was co-supervised by Prof. Akihiro Sugimoto, some experiment with pixel-wise segmentations in Section 5.6 was executed by Dr. Jan Svihlik.

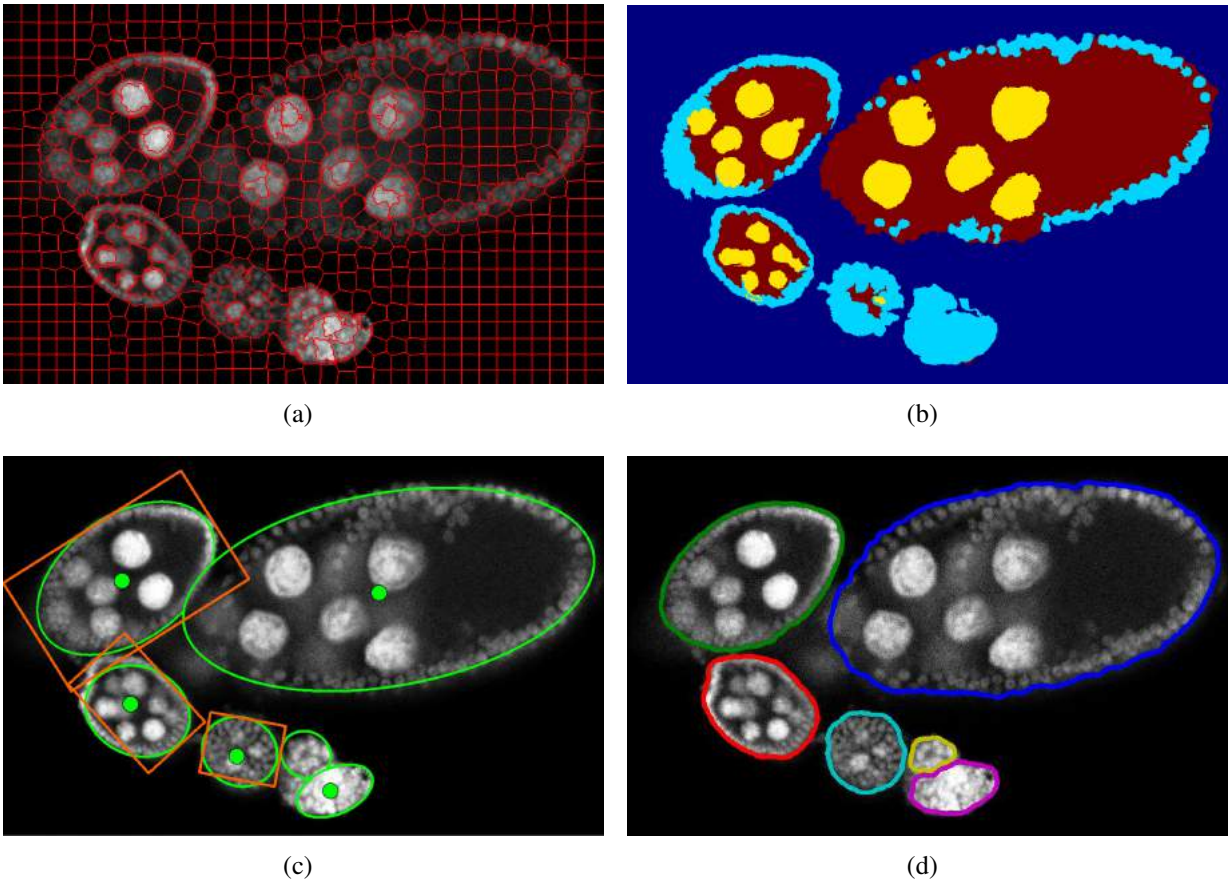
2
Background and State-of-the-Art

Image processing [69] is a vast field with many applications in a variety of areas from research to industry; from assisting tools for an expert to fully automatic systems; from the automotive sector to medical or biomedical at which we aim in this thesis. Overall, there are also many methods that can be successfully applied across several domains (applications) such as image segmentation, object detection and decomposition in a more general formulation.

In this chapter, we review the State-of-the-Art overview divided into three distinct areas, one for each contribution area of the thesis. First, the concept of superpixel and related works on superpixel segmentation (Section 2.1, 2.2) are presented. Second, an overview of region growing methods is given in Section 2.3, followed by an extraction of binary patterns in the context of medical imaging in Section 2.4.

2.1 Superpixels

The superpixels (supervoxels in 3D) [51,70] became very popular in last decade especially in medicine [56,71–77] since they allow for processing high-resolution images efficiently, and thus also for increasing the capture range (the size) of images to be processed (see Figure 2.1(a)). The image size is the crucial parameter regarding algorithm performance in many image processing and analysis tasks. Furthermore, the performance may depend not only linearly, but even exponentially on the input size, such as for Graph Cut [78] optimization. Using superpixel representation is a convenient way to group pixels with similar spatial (position) and temporal (color) information, and so they can be substituted by one superpixel. This allows to represent an image of 10^9 pixel in original resolution by much fewer superpixels, typically the drop is at least two orders, with comparable predictive value, (see in Figure 5.2). One of the key features of superpixels is producing compact regions with an ability to preserve object boundaries. Su-



(a) (b)

(c) (d)

Figure 2.1: Illustrative example of a fluorescence microscopy image containing several egg chambers in different processing phases. *(a)* Superpixels estimated from magenta channel visualised as red contours over input image. *(b)* Automatic semantic segmentation into four classes: Background (blue), cytoplasm (red), nurse cells (yellow), and follicle cells (cyan). *(c)* Object center detection: expert's annotation (red), detected object center and ellipse approximation (green). *(d)* Manually delineated egg chambers used for validation - instance segmentation.

perpixels are typically used in image segmentation [40,56,73,79,80], registration [81,82], object detection [83,84] but also in deep learning [85,86].

For a formal definition, let us consider an input image $I : \Omega \rightarrow \mathbb{R}^m$ with m color channels, which is defined over a pixel grid $\Omega \subseteq \mathbb{Z}^d$. A superpixel s denotes a group of pixels Ω_s and decomposition of image I onto a set of superpixels S is a parcellation of Ω into pairwise disjunct superpixels $\Omega = \bigcup_{s \in S} \Omega_s$.

There are many approaches to calculate superpixels — clustering [87], watershed [88], graph-based [89], etc., which usually provide a very similar image parcellations [51]. Results of two benchmark comparison studies showed that the method of choice is strongly dependent on the target application [51,70]. Furthermore, the choice represents always a trade-off between boundary recall and processing time, with no universal method reaching best results in both aspects.

In this work we chose Simple Linear Iterative Clustering (SLIC) [87] as the baseline superpixel method on its excellent trade-off between processing time and high boundary recall. The basic ideas behind SLIC are common k-means [90] clustering on joint spatial and temporal space with limited search space in the spatial domain. Over time, several implementations of SLIC have been introduced with some improvements in speed, for instance, using graphic card [91–93] and segmentation performance by considering minimum path costs between pixel and cluster centers rather than their direct distances [94], or using adaptive compactness factor determined according to the image neighborhood variance [95].

2.2 Superpixel segmentation

The image segmentation [45–47, 69] is a general problem that occurs in computer vision. Its task is to identify and split an image into regions with different context-dependent meanings, such as the four classes (each marked by a different color) shown in Figure 2.1(b) that corresponds to various biological tissues in the input image (Figure 1.2(a))

Formally, a segmentation function $\mathbf{y}_\Omega : \Omega \to \mathbb{L}$ assigns label $l \in \mathbb{L}$ to each pixel, where $\mathbb{L}$ is the application-specific set of classes (labels). In most segmentation tasks, one can assume that the regions of equally labeled pixels should be somehow compact, i.e., neighboring pixels share a label. In this sense, superpixels can be seen as an image segmentation but with a difference that the number of a label is much larger and there is just local meaning for each label. Following the previous definition, superpixel segmentation is then a function $\mathbf{y} : S \to \mathbb{L}$ assigning a label to each superpixel. Note, that the superpixel label can be simply propagated (distributed) to all pixels belonging to the particular superpixel.

Especially in medical imaging, many image segmentations based on superpixels have been presented [56, 71–77] mainly as a preprocessing phase in further image analyses [71], segmentation, registration [81], etc. The advantage of using superpixels except reducing the degree of freedom (number of variables) comparable image information is also higher robustness to noise as can occur for multiple modalities, typically US or MRI. It has been shown that superpixels lead to more accurate segmentation than direct pixel-level segmentations while having lower demands on processing time and memory resources.

Let us define the commonly used pipeline for superpixel segmentation which is very similar to much pixel-wise segmentation. Existing superpixel-based segmentation methods [56, 72, 75–77] usually consist of 3 steps: (i) computing superpixels, which preserve required details such as object boundary and reasonable superpixel size; (ii) extracting superpixel appearance features based on image intensity (e.g. color [71,73], texture [56,76]) or superpixel shape [72]; (iii) using an estimated model or classifier to assign a label to each superpixel. This classification can be performed by a standard classifier in a supervised [56, 72, 76, 77] or unsupervised manner [71, 73, 74, 96] with a fundamental assumption about the input images such as a number of classes or interclass differences.

Moving from independent pixel classification and employing spatial relations between neighboring pixels through Markov Random Fields (MRF) or some other graph-based regularization, e.g. GraphCuts [78,97–99] improved the segmentation results regarding accuracy and robustness

to noise, but at the costs of increased computational demands. The most similar previous work was done by Kitrungrotsakul et al. [80], the authors also use superpixels and Graph Cuts [78] as a regularizer, but their work focuses on a single application — single object binary segmentation. Punyiani et al. [71] uses a simpler superpixel extraction method and only basic edge terms in the graph regularization. Ye [100] uses mean shift clustering, and Hsu et al. [101] worked on SLIC superpixels with a region-adjacency-graph regularization, while Wang [102] considers long-range similarity-based interactions instead of interactions based on neighborhoods.

Finally, there seems to be a great promise in deep learning methods using convolutional neural networks, such as U-net [49] requiring a small number of training examples but it assumes reasonable homogeneous objects and produces only binary segmentation where proper object separation relays on exact boundaries prediction.

Our key contributions to superpixel segmentation with respect to [80] are as follows: (a) formulating the task as general multi-class segmentation; (b) proposing a new formula for the edge weights based on differences in model-based class probabilities; (c) incorporating both unsupervised and supervised modeling; and (d) including a comprehensive set of experiments that shows that our method can be applied to four different applications with little adjustment.

2.3 Region Growing

The region growing [103–106] together with level-set [107–109] methods form a special group of image segmentation methods. The region growing starts from a seed (point or a small region) and aggregate neighboring pixels or regions (superpixels) according to given rule which minimizes a global optimization criterion. These methods: (a) they typically require an initialization or starting point which can be set manually or estimated automatically for instance as local minima of some global criterion; and (b) the optimization is usually performed iteratively. Example of sample image and a region growing segmentation is shown in Figure 1.2(a) and Figure 2.1(d) respectively.

The watershed [110, 111] segmentation on distances to boundaries is a typical example of region growing method starting from local minima, and stopping at touching the neighboring region. In level-set segmentation, the object of interest is described by a function and the resulting segmented region corresponds to the zero-levelset. Multiple objects can also evolve at the same time, but policy definition on their intersections is much harder.

Ye [100] uses mean-shift superpixels and Graph Cuts, while the MRF optimization can also be solved by Gibbs sampling or simulated annealing [112, 113]. Unlike the present work, neither of these methods can handle multiple interacting objects and incorporate shape models.

As one major alternative, Graph Cuts can be combined at the pixel-level with a shape models such as the layered pictorial structures [114], the distance functions to a shape template [115, 116], or the star shape model [117]; it is also possible to choose between multiple models [118]. These methods alternate between estimating the pose parameters and refining the segmentation and can converge to a suboptimal solution if the pose estimation is incorrect. The number of pose hypotheses that can be simultaneously considered is limited for computational reasons. Global optimization concerning the shape model parameters is possible but very computationally

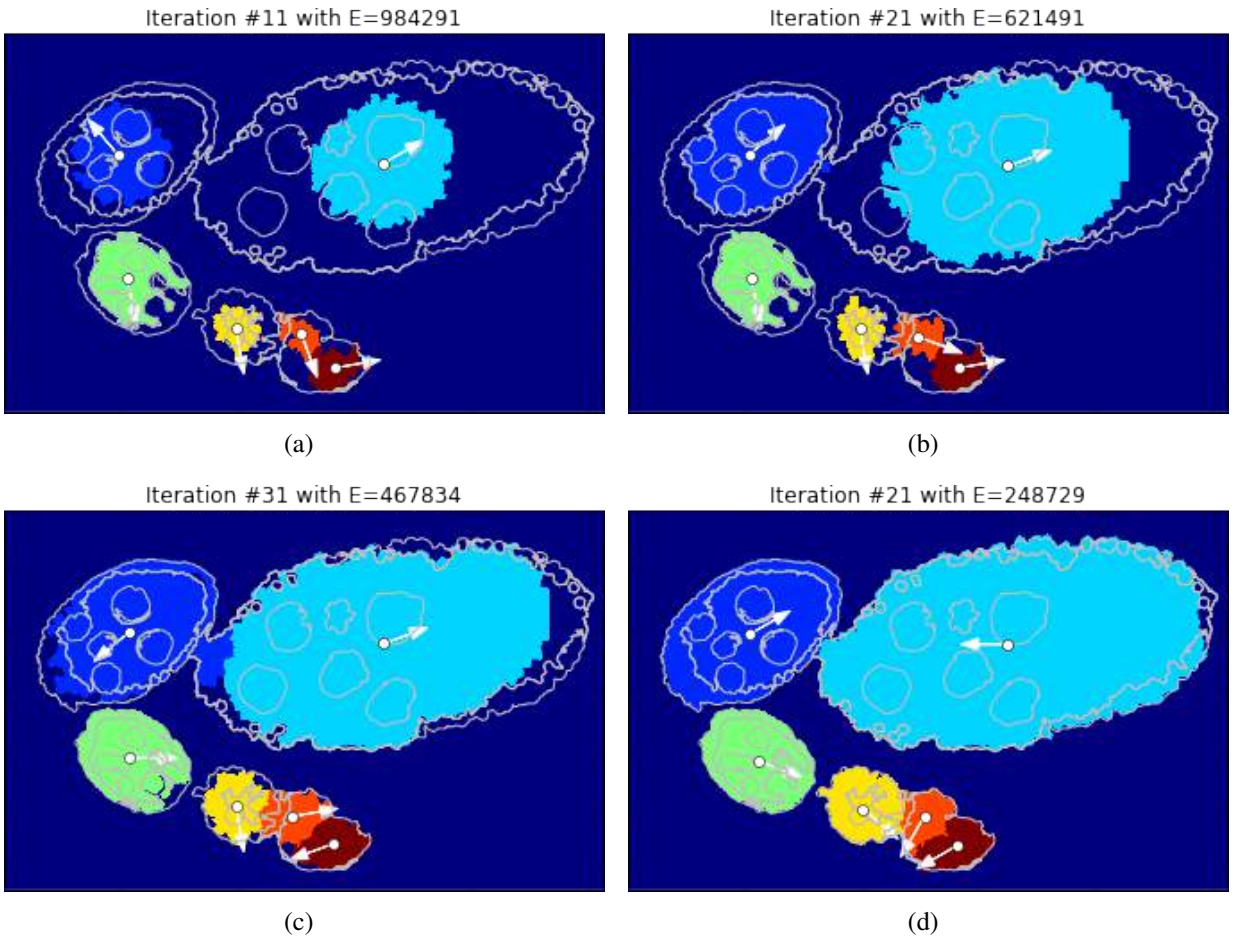


Figure 2.2: Several iteration steps region growing shown on Drosophila ovaries, Each color region corresponds to an individual object k, the thin white contours are class contours of semantic segmentation. White dots represent the centers c_k of mass and white arrows the principal orientations Θ_k for each object. For more detail see Chapter 7.

expensive [119]. Graph Cuts can also be augmented by constraints on class distances [120], one region being inside another [121, 122]. All these methods are slower than applying Graph Cuts on superpixels.

Region growing is similar to active contours [69, 123], which can be interpreted as region boundaries using region-based criteria [124], and are also often used in biomedical imaging, for example for cell segmentation and tracking [109]. Active contours can be used to segment multiple objects using, e.g., multiphase level sets [107] or multi-object active contours [125]. Objects may be allowed to overlap, or separation between objects can be enforced [125, 126]. Shape priors can be integrated using the usual alternative optimization of pose and segmentation [127–130]; specialized methods exist for simple shapes such as circles [131, 132]. Active contours can provide subpixel accuracy, but their computational complexity is often very high, although fast discrete methods exist [133, 134].

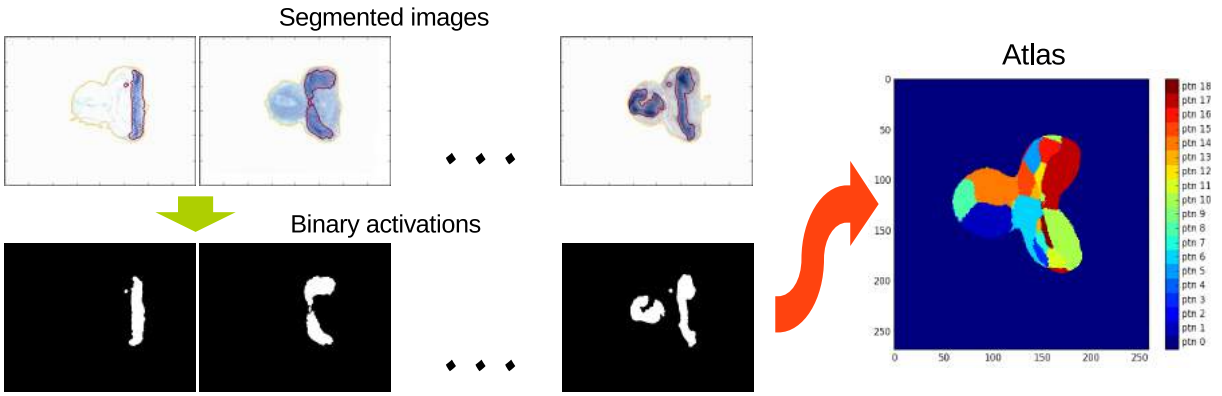


Figure 2.3: A simplified flowchart of extracting gene activation as binary segmentation from already aligned images according to segmeted objects (imaginal discs) and obtaining an atlas of compact non-overlapping patterns.

Our key contribution consists of three main aspects. First, we grow the regions based on superpixels instead of pixels (see Figure 7.4(b)), which improves the segmentation speed by several orders of magnitude while the superpixels preserve the object boundaries. Note that region growing by superpixels, i.e. representing regions using superpixels, is very different from calculating superpixels by region growing [135]. Second, we incorporate a shape model based on the so-called ray features [136, 137], to guide the region growing towards plausible shapes, using our a priori knowledge. Multiple alternative models can be used in parallel. Previously, shape models for region growing were described for example by a distance map [138, 139] or moments [140]. We build a probability model over the shape features using histogramming, other options include PCA, or manifold learning [141–143]. Third, our method segments several objects simultaneously, ensuring that they do not overlap. One iteration of the growing process is formulated as an energy minimization problem with a MRF regularization and solved either greedily or using Graph Cuts. Since the number of boundary superpixels in a given iteration is small, the procedure is very fast. In contrast, applying Graph Cuts to all superpixels [38, 144] is much more time and resources demanding.

2.4 Atlas extraction

The task of constructing an atlas of a finite number of universal locations [145] of gene activations which are mutual for all input images can be interpreted as decomposition of an image into a small number of compact regions (patterns). Such pattern extraction task (or similarly dictionary learning) belongs to a wider family of image decomposition methods. Consider a matrix $\mathbf{G} \in \mathbb{R}^{|\Omega| \times N}$ to represent by a rearranged set of pixels of N images with pixel coordinates Ω in one dimension (the spatial relation is dropped in this representation). The underlying assumption is that values in each position represent the same appearance. In other words, the images are assumed to be aligned, and the pixel values correspond to the same appearance model.

A linear decomposition of the matrix $\mathbf{G}$ can be found by minimizing

$$\min_{Y',W} \left\| \mathbf{G} - Y' \cdot W \right\|^2$$

where $Y' \in \mathbb{R}^{|\Omega| \times \mathbb{L}}$ corresponds to a dictionary (or 'atlas') with $\mathbb{L}$ patterns and $W \in \mathbb{R}^{L \times N}$ are image specific weights. (Note that our task is not linear but binary for patterns and their weights.)

We shall give a few examples of known methods, differing in additional assumptions and constraints. Built on the well-known PCA, sparse Principal Component Analysis [146] (SPCA) assumes the weights W to be sparse. Fast Independent Component Analysis [147] (FastICA) seeks for the spatial independence of the patterns. Dictionary Learning [148] (DL) with Matching Pursuit is a greedy iterative approximation method with many variants, mainly in the field of sparse linear approximation of signals. Non-negative Matrix Factorization [149] (NMF) adds the non-negativity constraints, while sparse Bayesian models add a probabilistic weights, encouraging sparsity. Both methods were used for estimating gene expression patterns in *Drosophila* embryos [25, 150] (see Berkeley Drosophila Genome Project[1]).

There is far less literature in the case of binary $\mathbf{G}$, Y_Ω, or $\mathbf{W}$, see schema in Figure 2.3. If the requirement of spatial compactness of the patterns is dropped, then the problem is called binary matrix factorization [151, 152] and is often used in data mining. Simplifying further to allow only one pattern per image leads to the problem of vector quantization [153].

Regarding the binary input $\mathbf{G}$, sparse logistic PCA method was introduced in [154], also called regularized PCA, for analyzing binary data by maximizing a penalized Bernoulli likelihood using Majorization–Minimization algorithm for minimizing of the negation form. Moreover, online dictionary learning from binary data [155] aims at reliable reconstruction while penalizing the sparse representation vector.

In the medical imaging domain, a similar problem is to estimate 'atomic' activation patterns from fMRI images from brain fMRI images. It was solved for example using sparse-structured PCA (SsPCA) with l_1 and l_2 norm regularization. The problem was later extended to the multi-subject case [156]. Compact activation patterns were encouraged by the total variation criterion [157]. In all these cases, the problem was solved in the continuous setting, with binary patterns Y_Ω obtained by thresholding if needed [157].

Some automatic methods exist, based on, e.g., staining level extraction based on GMM Decomposition [23, 158], loopy belief propagation to find the maximum a posteriori for being a pattern as part of SPEX2 [71], sparse Bayesian factor models [25] or non-negative matrix factorization [150]. In contrast to these methods, we assume in this work that the activation and the patterns are inherently binary. We further assume that the patterns, corresponding to anatomically defined zones, are compact and non-overlapping. These constraints should increase the robustness of the estimation and yield more biologically plausible results.

[1] http://insitu.fruitfly.org/cgi-bin/ex/insitu.pl

3 Datasets and materials

We shall compare the performance of our methods with existing algorithms for two applications described below. In this Chapter, we first introduce the real datasets and their usage in biomedical imaging, then synthetic datasets used for atlas extraction. Second, we describe the evaluation metrics used for each particular task — image segmentation, instance segmentation and dictionary learning.

These datasets have been particularly introduced in papers [38–40, 44, 68]. For clarity and compactness of the thesis, we unify their description and remove some other biomedical images used in published paper [40], for instance, in histology tissue and Langerhans islets.

3.1 Real microscopy images

Drosophila melanogaster, the fruit fly, is often used in biological research, due to its high genetic similarity to humans [2,17,26], its rapid evolution, and its short life cycle. These features allow genetic changes to be observed easily across generations.

Note, that the used images are the same for multiple application, but we had to adjust the reference segmentation to suit the particular task such as semantic or instance segmentation.

3.1.1 Drosophila imaginal discs

Imaginal discs are parts of the larvae from which the adult body parts develop, see Figure 1.1(c). The expression of about 6000 genes was imaged by RNA in situ hybridization [14]. Each gene was imaged a number of times for four imaginal disc types, with the aim of determining the role of the gene in the development process [15]. The disc anatomy is represented by the gray-scale part of the image, with the gene expression in blue (Figure 3.1). Segmentation of the

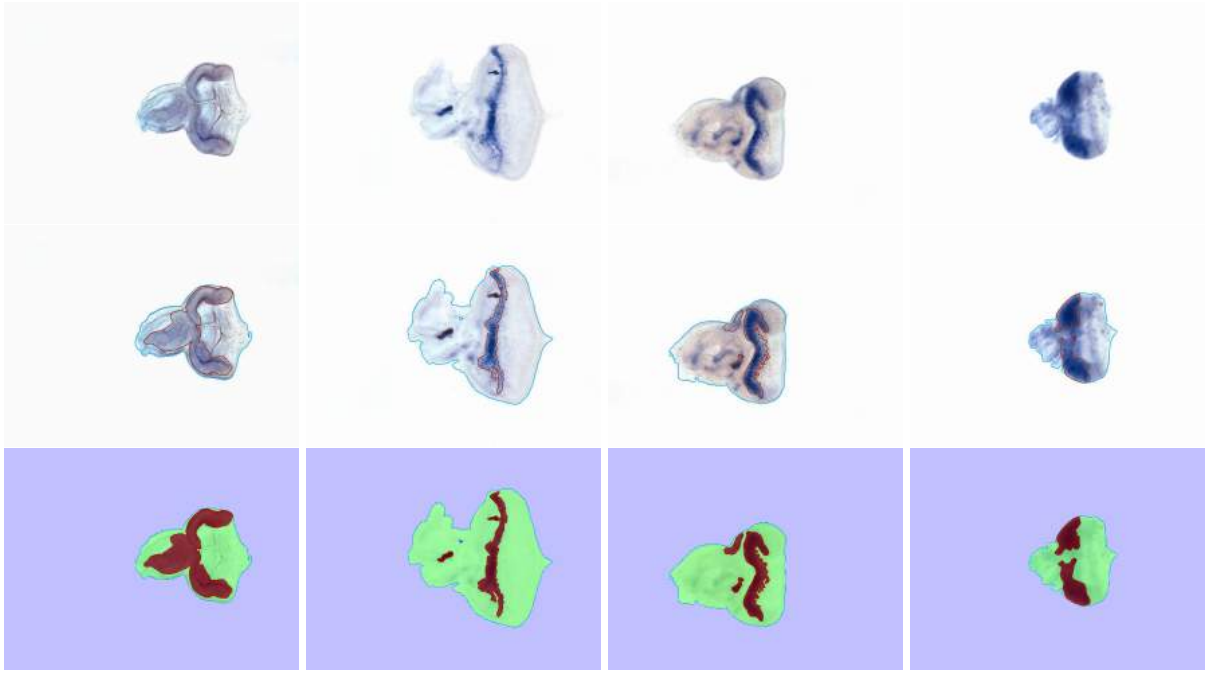


Figure 3.1: Example of microscopy images of Drosophila imaginal discs *(top row)* and manual segmentations *(bottom row)*, with the gene activation class colored in red and the rest of the imaginal disc colored in green. In the middle row, the class contours are shown as red and blue lines, superimposed over the original images.

disc is needed for the subsequent analysis, which consists of aligning all discs of the same type, detecting the activations and processing them statistically [44].

This dataset (see samples in Figure 3.1) contains several thousand images of imaginal discs of 4 types – wing, leg, eye and haltere, respectively. The evaluation is done on 15 images of imaginal disc type 3 (eye), for which we have reference segmentations. These annotated images cover most of the appearance variability in the dataset, and initially, only validation of unsupervised segmentation was assumed. Let us note, that many medical applications suffer from lack of annotations.

The resulting unsupervised segmentation of imaginal disc introduced in Chapter 5 serves as input binary images for dictionary learning (see Chapter 8). In Figure 3.2 we present several segmentations consecutively: the segmentation of disc and gene activation classes overlayed on the input image, disc segmentation (union of disc class and gene activation) which is used to register discs to a common disc prototype, and gene activations used for atlas extraction.

3.1.2 Drosophila ovaries

Drosophila ovaries were imaged in 3D using fluorescence microscopy and fluorescently labeled RNA probes, in order to study gene expression patterns during Drosophila oogenesis, see Figure 1.1(b). The images contain two channels, cellular anatomy and gene expression patterns.

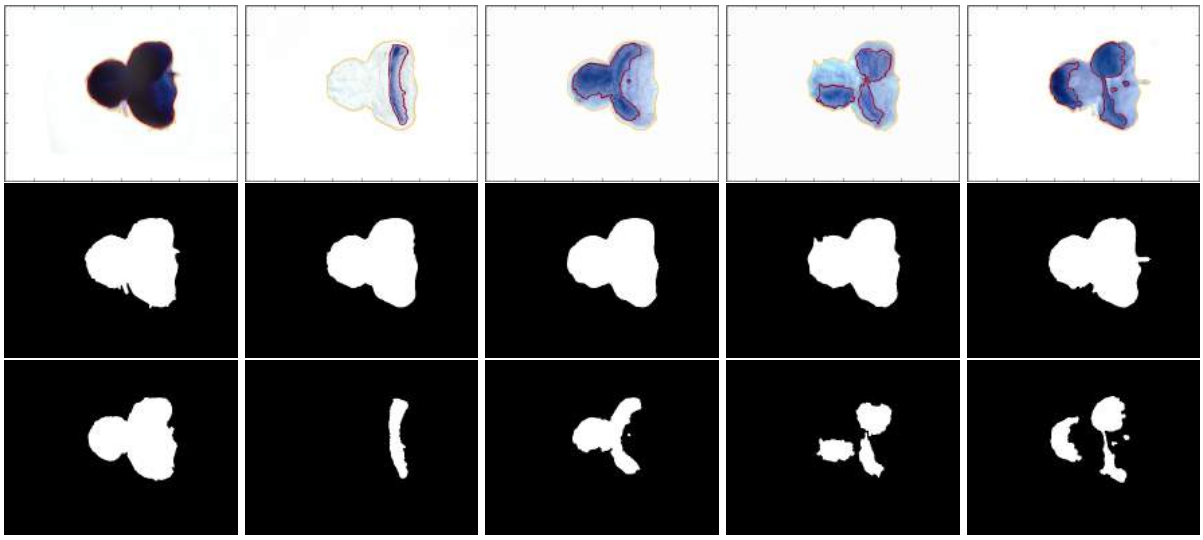


Figure 3.2: Presenting samples of aligned Drosophila imaginal discs (eye antenna discs type) and their segmentation. First, we show the sensed images *(top row)* with marked contour of segmented disc (orange) and gene expression (red) followed by visualization of the segmented discs *(middle row)* and segmented gene expressions *(bottom row)*.

Only the cellular anatomy channel is used for semantic and instance segmentation. Almost 20 thousand 3D volumes were acquired, from which the experts extracted relevant 2D slices, each of them typically containing a few eggs, linked into a chain (see Figure 1.2(a) and 2.1). Note that we used the same input images in several tasks and so the annotation also differs — semantic used in Chapter 5 and instance annotations used in Chapter 7 and 6.

Semantic (tissue) segmentation

We have reference segmentations for 75 cropped images of individual eggs and 72 additional complete slices where a user manually annotated the four tissue classes: follicle cells, nurse cells, cytoplasm and background, for all development stages (see samples in Figure 3.3).

This segmentation, as well as the fine segmentation of our superpixel segmentation methods (see Chapter 5), is later used for training object center detection with label histogram features.

Instance (individual egg) segmentation

We utilize only the images of whole 2D slices. We start with semantic annotation introduced in Section 3.1.2 and we merge all tissue classes (see Figure 2.1(b)) as foreground and manually split touching eggs by thin backgrounds obtaining individual objects (see Figure 2.1(d)). For 72 images, containing approximately 250 eggs, we have a full pixel-level manual segmentation.

Reference segmentation of individual eggs, also called instance segmentation, is used for training and validation of object center classifier (see Sec 6.2) and for learning shape prior for region growing together with evaluation of segmented objects (see Section 7.1.3).

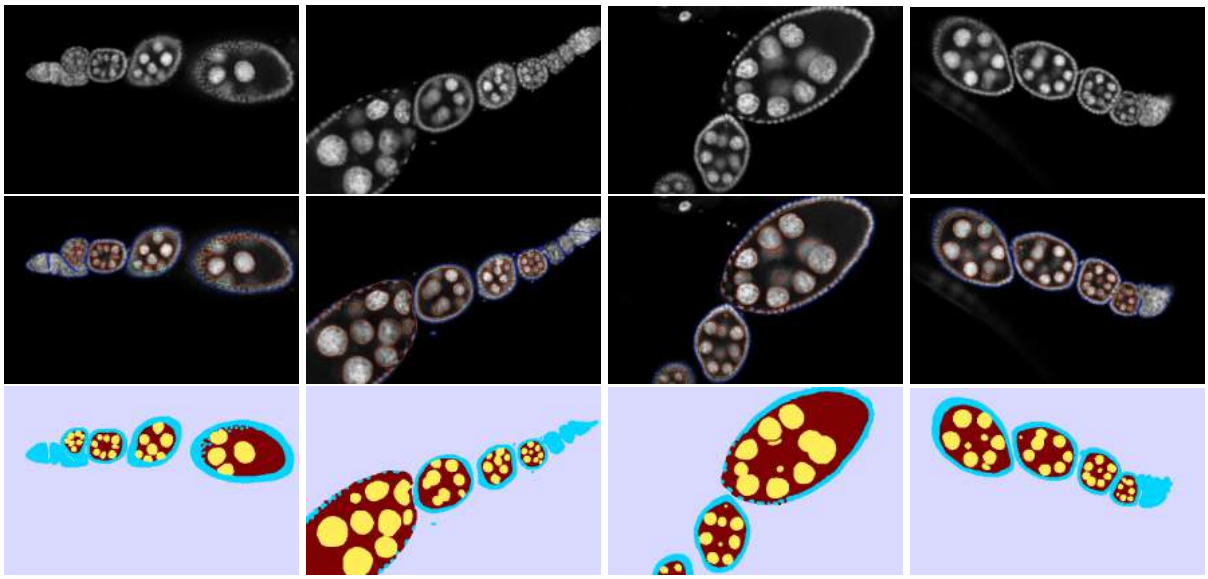


Figure 3.3: Examples of microscopy images of Drosophila ovaries *(top row)*, manual segmentations of individual eggs into 4 classes *(bottom row)* — follicle cells (cyan), nurse cells (yellow), cytoplasm (red) and background (blue). The same segmentations visualized as contours superimposed over the original images *(middle row)*

Ovary egg detection

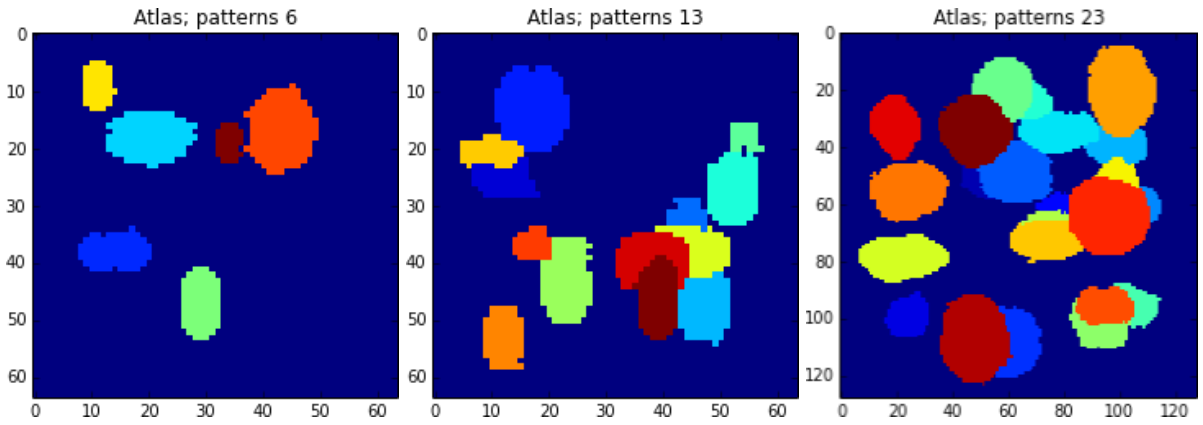
Drosophila egg chamber development can be divided up into 14–15 stages [7, 41]. However, some of them are hard to distinguish, and the differences are not relevant to this study. For this reason, our dataset recognizes only five developmental stages, numbered 1–5 that correspond to stages 1, 2–7, 8, 9, and 10–12 of [41], respectively. Stage 1 in our notation corresponds to the smallest egg chambers without any distinguishable internal structure (e.g., the smallest egg chamber in Figure 2.1(d)) and it is no interest for gene expression analysis.

The validation dataset consists of 4338 2D slices extracted from 2467 volumes. Experts identified the developmental stage and approximate location for 4853 egg chambers by marking three points on their boundaries (begin and end position of the egg chamber and marking the width in the most extensive section along main diagonal). In Figure 6.6 we represented these three points by a bounding box shown as red rectangles. Note that all development stages are presented approximately equally (see Table 6.6)

This dataset is used only in Section 6.4.2 for presenting detection rate and a positive impact of ellipse fitting on multiple center detection inside a single egg.

3.2 Synthetic dataset - binary patterns

The expert annotation of an atlas of gene activation is very sparse, and typically contains only four locations — top, bottom, front and back of the ovary chamber. It is clear that for



(a) v0 (b) v1 (c) v2

Figure 3.4: Visualization of the generated atlases for the three created synthetic datasets containing $K = 6$ *(a)*, $K = 13$ *(b)* and $K = 23$ *(c)* patterns, with image sizes 64×64 *(a,b)* and 128×128 *(c)* pixels.

proper evaluation of estimated atlases it is not sufficient, so we created several synthetic datasets simulating the observation from real images. Using such datasets allows us to measure more quality parameters. These datasets are used in Chapter 8.

We generated three synthetic datasets (v0, v1, v2) representing already segmented and aligned binary images based on a random atlas and random pattern weights. These datasets differ in image size, number of used patterns and complexity where we assume the v0 as the simplest and v2 as the hardest (see Figure 3.4). The patterns are deformed ellipses.

Each dataset is further divided into three sub-sets, each containing 1200 images (Figure 3.5):

1. **pure:** images generated from equation (8.2)

2. **deform:** pure images (from point 1.) independently transformed by a small elastic B-spline deformation with the maximum amplitude of $0.2\sqrt{|\Omega|}$.

3. **deform & noise (D&N):** deformed images (2) with random binary noise (randomly flipping 10% pixels).

With the addition of deformation and noise, we also increase the difficulty level and at the same time we imitate the real biomedical images which usually suffer from these issues.

3.3 Evaluation metrics

In this section, we introduce the used metrics for performance evaluation of all methods on previously presented datasets.

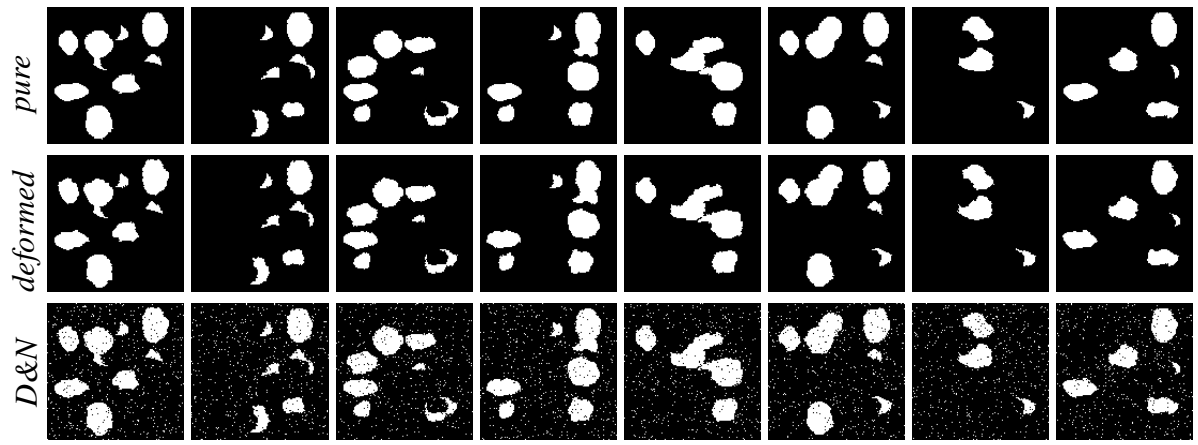


Figure 3.5: We show sample images from the synthetic dataset v2 (see Figure 3.4). The three rows represent the 3 sub-sets of input images: pure, deformed and deformed with random binary noise (denoted D&N).

3.3.1 Segmentation quality criteria

Regarding the semantic (see Section 5.6), we use standard evaluation measures for multi-class classification [159] - F_1 score, accuracy, precision, and recall. We also use the adjusted rand score or index (ARS) [160], which is the corrected-for-chance version of the Rand index, which is a statistical measurement of the similarity between two segmentations (groupings) without exact labels matching. The ARS counts the number of times that two objects are classified into the same class or different classes by the two methods and it can yield values in the range of $(-1, 1)$, the higher, the better.

For evaluation of the instance segmentation performances (see Section 7.3) we use the standard measures [159] plus Jaccard index — computed on the binary object/background results.

3.3.2 Object center detection

Standard metric [159] measures the performances of a trained classifier - AUC, F_1 score, accuracy, precision, and recall. The automatic detection is considered as correct if the detected center is inside the user annotated bounding box (see Section 3.1.2 and Figure 6.4.2).

3.3.3 Binary pattern extraction

The design and objective comparison for the task of atlas estimation is truly connected to our task and its biological application — 'atlas' estimation as a set of compact patterns.

Atlas comparison, we can see the estimated 'atlas' as a multi-label segmentation with a priory unknown number of classes and without giving a particular relation between (among) classes in estimated segmentation and desired atlas. The difference between atlases can be measured by

the ARS [160], which is commonly used in clustering measures and gives similarity value in the range $(-1,1)$, with 1 being a perfect match.

Reconstruction difference. Regarding the real image, we do not have a ground-truth segmentation as we do not know the ideal number of patterns in the 'atlas'. We decided to measure the expressiveness of the estimated 'atlas' back reconstruction error from estimated atlas and encoding over all images. Here, we refer the motation introduced in Section 2.4 and further developed in Section 8.2. With the estimated atlas $\mathbf{y}_\Omega$ and pattern weights $\mathbf{w}^n \in \mathbf{W}$ for each particular image $g^n \in \mathbf{G}$ we reconstruct each input image $\hat{g}^n$ (see Chapter 8, equation (8.2) and Figure 8.3), the approximation error is averaged over all images:

$$R(\mathbf{G}, \mathbf{y}_\Omega, \mathbf{W}) = \frac{1}{N \cdot |\Omega|} \sum_n F(g^n, \mathbf{y}_\Omega, \mathbf{w}^n) = \frac{1}{N \cdot |\Omega|} \sum_n \sum_i |g_i^n - \hat{g}_i^n| \tag{3.1}$$

Note, in case of the synthetic datasets we always compare the reconstructed images to the pure input images.

4 Superpixel extraction

In this chapter, we briefly review the SLIC superpixel algorithm as it was originally proposed in [87, 161] and introduce some improvements in performances. Since many image applications which use superpixel as their critical building block, they require high boundary recall and rely on fast processing time. We reformulate the regularization parameter to be more invariant to superpixel size and we propose some other 'algorithmic/implementation' improvements that speed-up the superpixels extraction. We noticed that original SLIC has an unsolved issue in post-processing phase where too small regions are joined to neighboring larger region to form superpixels. We propose smarter way bow to decide whether to merge regions or to keep them split base on their internal properties. The superpixels are used as a building block for all following methods except the BPDL.

This chapter is strongly based on the paper [68]. Compare to the original text; some notations were changed to unify with the rest of the thesis.

4.1 Simple Linear Iterative Clustering

The superpixel extraction is an over-segmentation method which group neighboring pixels with similar spatial properties. Formally, having an input image $I : \Omega \to \mathbb{R}^m$ defined on a pixel grid $\Omega \subseteq \mathbb{Z}^d$ we group pixels into superpixels, $\Omega = \bigcup_{s \in S} \Omega_s$, where Ω_s denotes pixels belonging to a superpixel s and S is a set of all superpixels (see Figure 2.1(a) or 4.1).

One of the widely used superpixel methods is SLIC [87, 161] which also has several reinterpretation targeting the processing speed [91, 93, 95] and boundary recall [94, 95]. The SLIC [87] is an adaptation of the k-means [90] algorithm for superpixel generation with two important distinctions:

1. the weighted distance measure

$$D = \sqrt{d_c^2 + \left(\frac{d_s}{\nu}\right)^2 \eta^2} \tag{4.1}$$

   combines color d_c (using the CIELAB color space, which is widely considered as perceptually uniform for small color distances) and spatial proximity d_s

2. the search space is reduced by limiting to a region $2\nu \times 2\nu$, proportional to the superpixel size ν

The search space reduction has a great impact on the speed of the whole algorithm, resulting on a complexity of only $O(N)$ instead of $O(KN)$ for standard k-means in each iteration, where $N = |\Omega|$ is the number of pixels in an image, and $K = |S|$ is the number of clusters [161]. Note that local optimization as it is proposed in SLIC requires significantly less iteration than standard clustering over the whole image, the authors claim that in average up to 10 iteration of SLIC is sufficient. The optimization is performed as an alternation between two steps — distance update and label assignment until it converges.

4.2 Regularization constant

The SLIC contains a regularization parameter η which influences the compactness of clustered superpixels. This constant η weights the spatial distance d_s and it is expressed as $\left(\frac{\eta}{\nu}\right)^2$ from equation (4.1) where (according to the notation in [87]) ν is the initial superpixel size and η is a parameter related to the maximal color distance N_c. We propose instead to use a parameter ξ defined in the range $\langle 0, 1\rangle$, where 0 means the minimal and 1 the maximal compactness.

$$\left(\frac{\eta}{\nu}\right)^2 = \nu \cdot \xi^2 \tag{4.2}$$

In our experiments, we found that the optimal default regularization value $\xi = 0.2$ works well for most cases. It is a good compromise between the superpixel compactness and fitting boundaries of the expected object in an image.

4.3 Implementation and speed-ups

Several implementations of SLIC already exist in multiple languages and frameworks. The author of [87, 161] provides a C source code which was wrapped into Python. Other implementations can be found also for Matlab (VLFeat library [162]). For real-time computer vision problems, SLIC has also been transformed to be fully processed on graphics cards with some minor improvements as gSLIC [91, 92].

Here we implement SLIC as a plug-in for ImageJ in Java. We are trying to maintain maximum compatibility with ImageJ. We used the ImageJ API as much as possible. We also had to use the native Java structures a few times to keep the clustering process as fast as possible.

4.3.1 Using Look-Up Tables

We analyzed the possibility of using precomputed Look-Up Tables (LUTs) to avoid repetitive computing of the same distances d_s in equation (4.1) or converting the same colors again. We found that we can achieve significant speed-up in specific cases (especially for color conversion) mentioned below.

Spatial distance in regular grid. The metric used in SLIC clustering contains a proximity distance

$$d_s = \sqrt{\left(x_j - x_i\right)^2 + \left(y_j - y_i\right)^2}$$

where $[x_i y_i]$ and $\left[x_j y_j\right]$ are coordinates of the cluster center and a pixel respectively. In a regular image grid, these distances are the same for all cluster centers and its proportional subset of neighboring pixels. Using this precomputed distance LUT with coordinate differences as entry; we gain a 5% speed-up.

Color conversion. Most commonly used images are in the RGB color space, while we compute the color distance in the CIELAB color space. It means that each image needs to be converted from RGB to CIELAB which is quite time-consuming. We found that the number of unique used colors in images is usually smaller than the number of pixels in the image. Typical images have only about 50% unique colors/pixels (e.g. Lena with size 512×512 pixels). For medical images, the ratio is even smaller. For example, a typical image of a stained histological tissue (see Figure 4.3*bottom*) contains less than 5% of unique colors/pixels. So we perform the conversion only when it is needed, so each used color is computed just once. It gives us a speed-up of about 60%.

4.3.2 Multi-threading

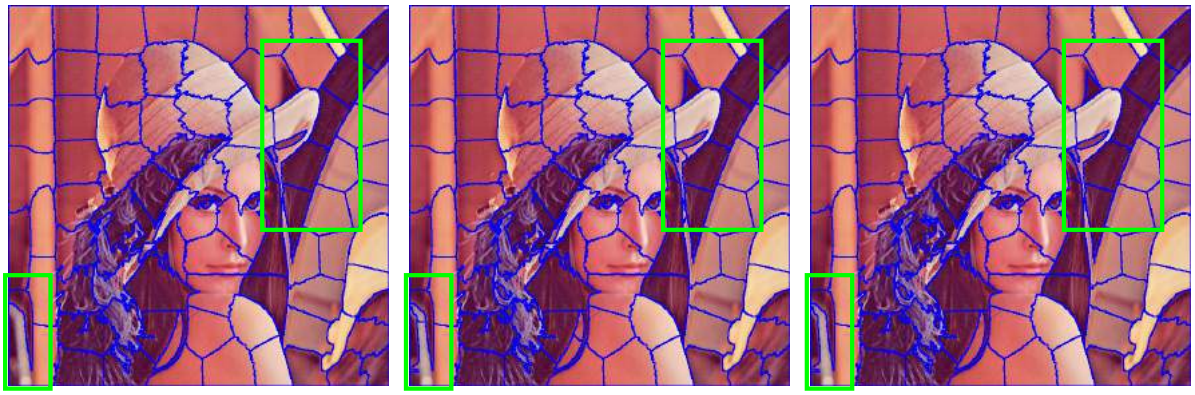
As the clustering is computed locally (for each superpixel only in its $2\nu \times 2\nu$ neighborhood, is quite simple to split the process by subsets of superpixels and/or image blocks into independent threads in both phases (standard *k*-means label assignment and distance update).

We apply the parallelism in the main loop of each phase - in the assignment phase each thread takes only a subset of all superpixels/clusters, and the update is computed per image blocks, such that each thread processes one image block.

The experimental results are reported later in Section 4.5.

4.4 Post-processing of outliers

The SLIC clustering may generate unconnected components (multiple regions which share the same superpixels label *s* but they are not connected together). The number of unconnected regions depends on superpixel compactness but in average (for regularization $\xi = 0.2$) there are



(a) (b) (c)

Figure 4.1: We compared the original SLIC condition for merging unconnected components $s^s < \varepsilon$ applying two different thresholds — original $\varepsilon = 0.25$ in (a) and decreased $\varepsilon = 0.06$ in (b). In (c) we introduced also our condition for merging described in equation (4.5). Original SLIC (a) just holds one large superpixel comparing to (b, c) which reasonably adds one more superpixel. On the other hand (b) adds some other small superpixels in nearly homogeneous areas, while (c) holds still single superpixels.

about $3 \cdot |S|$ unconnected regions where $|S|$ is number of expected superpixels depending on the image size and the initial superpixel size ν.

At first, all connected components z have to be found. We use a breadth-first search to compute all independent components z (for a 4-neighborhood). Then, for each component z we find a set of neighboring components ∂z. This early stage is the same for the original SLIC as it is for jSLIC post-processing.

Original SLIC post-processing [87]. The authors measure the relative area $z_a = \frac{|\Omega_z|}{\nu^2}$ of each component and merge small components if $z_a < 0.25$. For relabeling they simply use the label $z \leftarrow s$ of the first component from ∂z.

We found out that this simple approach is not optimal (see Figure 4.1), because some unconnected components are merged to superpixels even they have very different color and they are more similar to another neighboring superpixel, or introduce them as new superpixels. The authors, in their follow up work [72], deal with this issue by estimating smaller superpixels and setting the superpixel size smaller than the smallest detail in the image that they want to distinguish.

Proposed jSLIC post-processing. We propose a different post-processing step which takes into account all surrounding components ∂z and their similarity by color and area. We compute mean colors z_c in LAB color space and relative area z_a for all components. Then, we find the most similar component $z^* \in \partial z$ by computing the difference $d_z(z')$ between the mean colors of the components z and $z' \in \partial z$, and choosing the closest component z' with minimal distance in

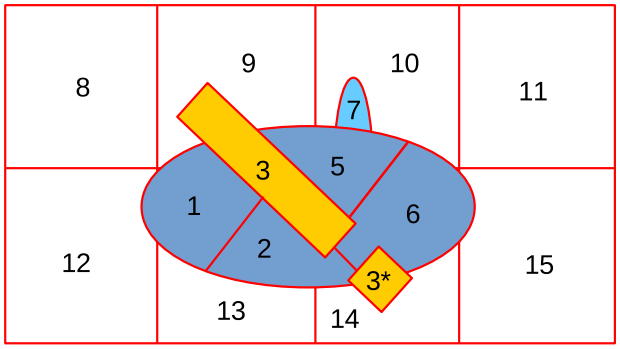


Figure 4.2: Sample scenario with extracted regions z (superpixels prototypes) before post-processing step. There is one unconnected region 3 which is a result of a single cluster but split by other regions 2 and 6. Example of proposed postprocessing, Tthe original SLIC would merge 7 to 10 because it is too small, compare to jSLIC which merges it to 5 because of the high color similarity. Similar, the region 3^* would be merged to 6 by original SLIC compare to jSLIC which would set it as independent superpixel 4.

color space

$$d_z(z') = \frac{||z_c - z_c'||_2}{z_a'} \tag{4.3}$$

$$z^* = \arg\min_{z' \in \partial z} d_z(z') \tag{4.4}$$

where the $||\cdot||_2$ is the Euclidean distance.

We experimented with the SLIC relabeling condition for unconnected components (see Figure 4.1). We found the original $z_a < \varepsilon$ condition insufficient even with various threshold values ε, because it does not take into account the color similarity. We propose a condition which solves this problem - the unconnected regions (see Figure 4.2) are merged if

$$\left(\frac{z_a}{4}\right)^2 \cdot \left(1 + d_z(z')\right) < \varepsilon \tag{4.5}$$

where $\frac{z_a}{4}$ expresses the relative superpixel size to the maximal superpixel size $2v \times 2v$. Experimentally, we set the threshold $\varepsilon = 0.25$.

4.5 Experiments

In this section, we compare performances of proposed enhancements regarding processing speed and more representative superpixels thanks to proposed post-processing step.

4.5.1 Performance speed-up

We perform this parallelization on a computer with 8-cores, and the results are presented in Figure 4.4(a). The most significant speed-up is between the single and 4-thread version. You can

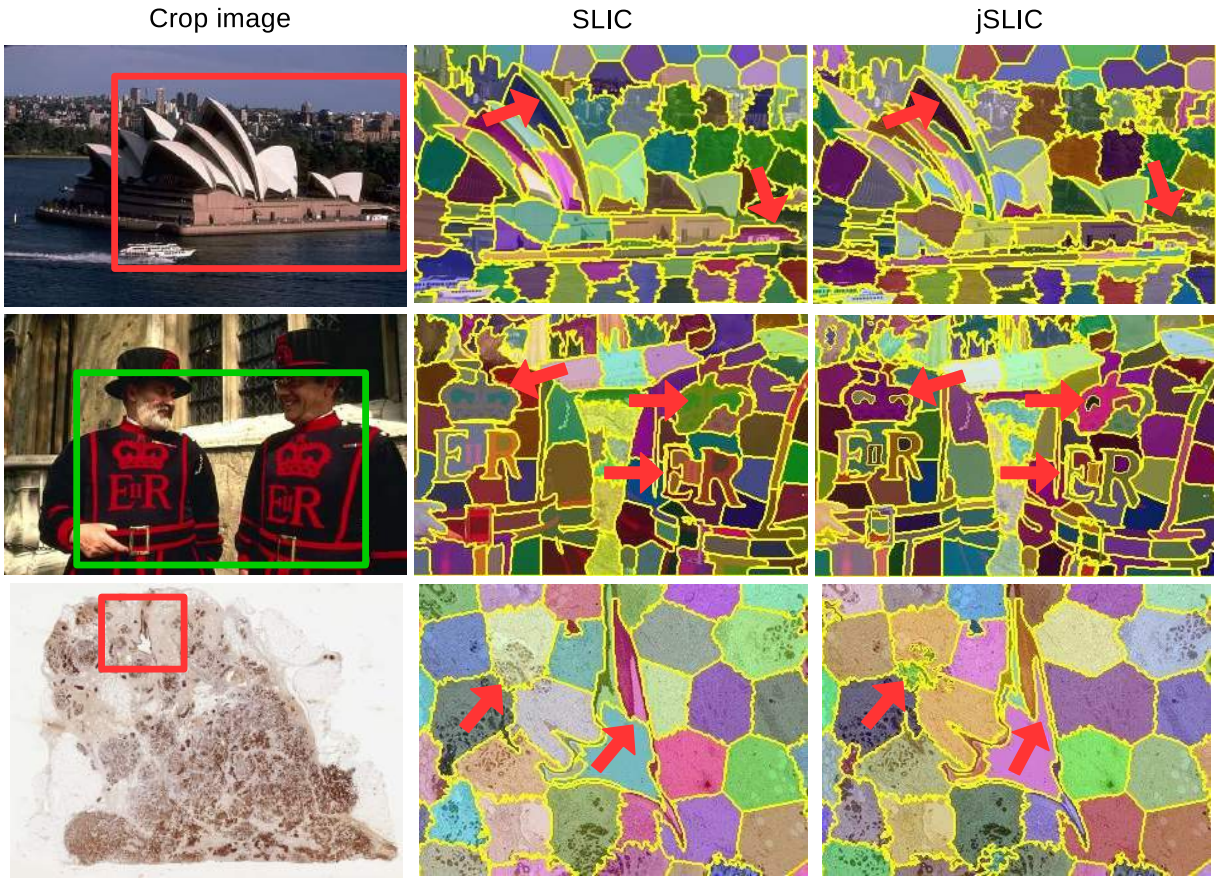
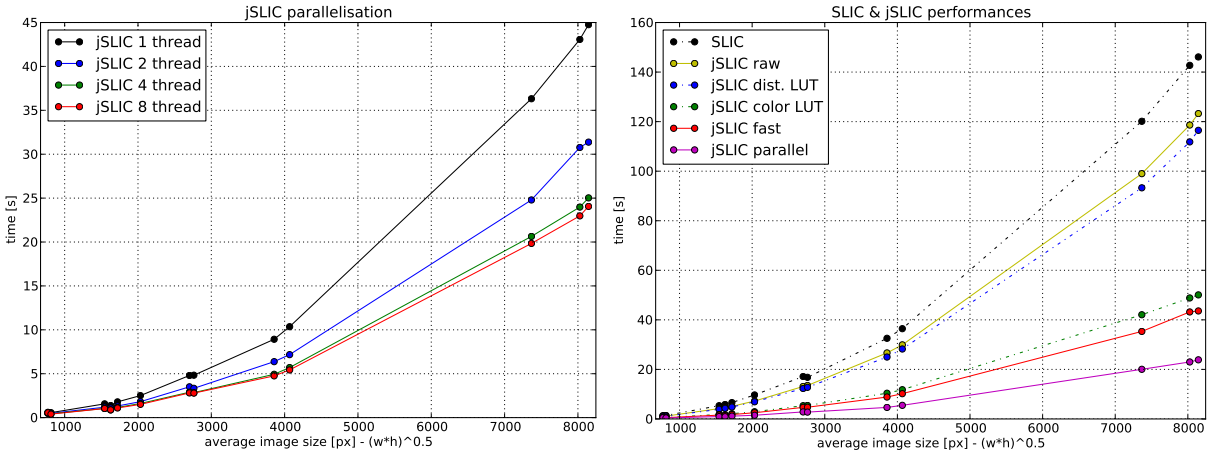


Figure 4.3: We run the original SLIC (middle) and jSLIC (right) on the Berkeley Segmentation Dataset [163] and some stained histological images using the same configuration for both. To present the differences we chose from each image only a part/detail where improvements can be easily seen (the rest of the image is usually segmented equally). The reason for more reliable superpixels by jSLIC is because it takes into account all neighboring connected components and their similarity by color.

see that the 8-thread version for small images takes even more time than 2-thread which is due to multi-threading overhead.

Table 4.1 presents the speed-ups of each proposed procedure. All following ratios are mean value over several histological images with different image size (see Figure 4.4(b)) and they express the relative speed-up to the original SLIC. Even without any of the improvements described in Sec 4.3, the jSLIC (implementation according [87]) is about 27% faster than the original SLIC implemented in C. Later the pre-computation of distances and converting each color just once brings 5% and 58% speed-up respectively comparing to the initial jSLIC and about 64% both together. In the end, the parallelization for 4-threads gives another speed-up of 37% to the fast jSLIC.

We applied jSLIC on several histological images of various image sizes, up to about 8.000×8.000 pixels, on a standard computer with a 4-core processor and 8Gb RAM. As a reference,



(a) (b)

Figure 4.4: The chart *(a)* shows benchmark results (elapsed time) on several histological images using 1-8 threads. The chart *(b)* presents the time dependency of complete superpixel clustering by SLIC and different variants of jSLIC depending on the number of pixels in the image. On average, the parallel jSLIC is 6 times faster than the original SLIC implementation.

we used the original SLIC implementation in C and compared it to our jSLIC in Java. The time dependency of all partial speed-ups on image size is presented in Figure 4.4(b). On average, we found the parallel jSLIC to be 6 times faster than the original SLIC implementation.

The experiments with parallelism show that the jSLIC is optimal when using up to 4-threads. Using more threads due to the threading overhead does not bring more significant improvements in performance.

4.5.2 Post-processing

For the evaluation of the post-processing step, we used a few images from the Berkeley Segmentation Dataset [163] and some stained histological images (see Figure 4.3). We made a visual evaluation of segmented superpixels concerning the amount of detail extracted from a given image. For both methods we set the same configuration - the same initial superpixel size $\nu = 30$ and regularization constant $\xi = 0.2$. To present the differences, we chose a detail in each image where the improvements can be easily seen (the rest of the image is usually segmented equally).

The advantage of the jSLIC post-processing is the ability to segment also smaller details than the initial superpixel size ν (see Section 4.4). We benefit from this fact when segmenting large histological images, where a big reduction of problem complexity is needed. For instance, have a look at the sample of a histological image (Figure 4.3 bottom), where the jSLIC is capable of estimating the hole in the tissue comparing to original SLIC method.

Figure 4.5 shows the influence of the superpixel size ν and regularization parameter ξ on the Drosophila imaginal disc images. Superpixel size ν is a trade-off between computational

method	speed-up
original SLIC	0%
jSLIC initial	27%
spatial proximity LUT	34%
color conversion LUT	217%
jSLIC fast (distance & color)	265%
jSLIC parallel (4 threads)	495%

Table 4.1: The table presents the relative speed-ups of each proposed procedure as a mean value over several histological images, and they express the relative speed-up with respect to the original SLIC.

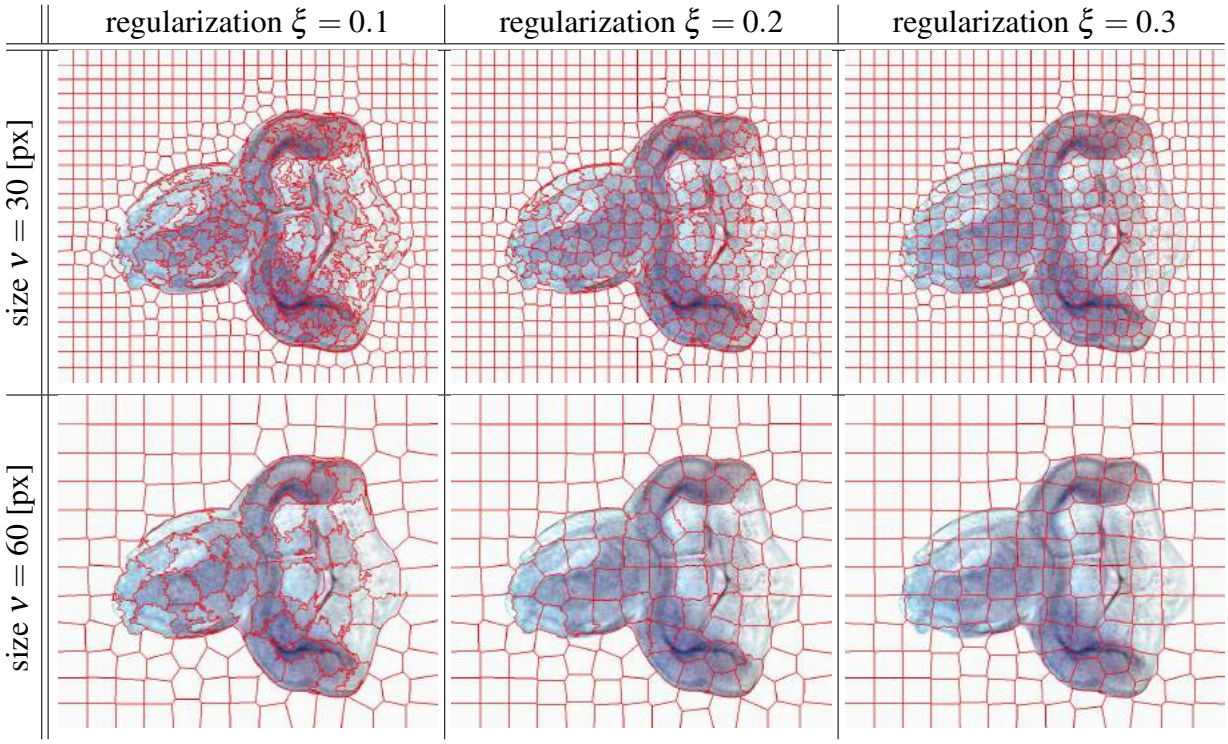


Figure 4.5: The influence of the two SLIC parameters on Drosophila imaginal disc: superpixel size and regularization.

complexity and the approximation error; values around $\nu = 25$ seems to work well for our data. Regularization ξ has a strong effect on superpixel regularity. Small values ($\xi \sim 0$) make the superpixels follow the contours in the image, with few shape constraints, while strong regularization ($\xi \sim 1$) makes superpixels mostly rectangular, regardless of the image. We observed that $\xi = 0.2$ provides good results for all our images.

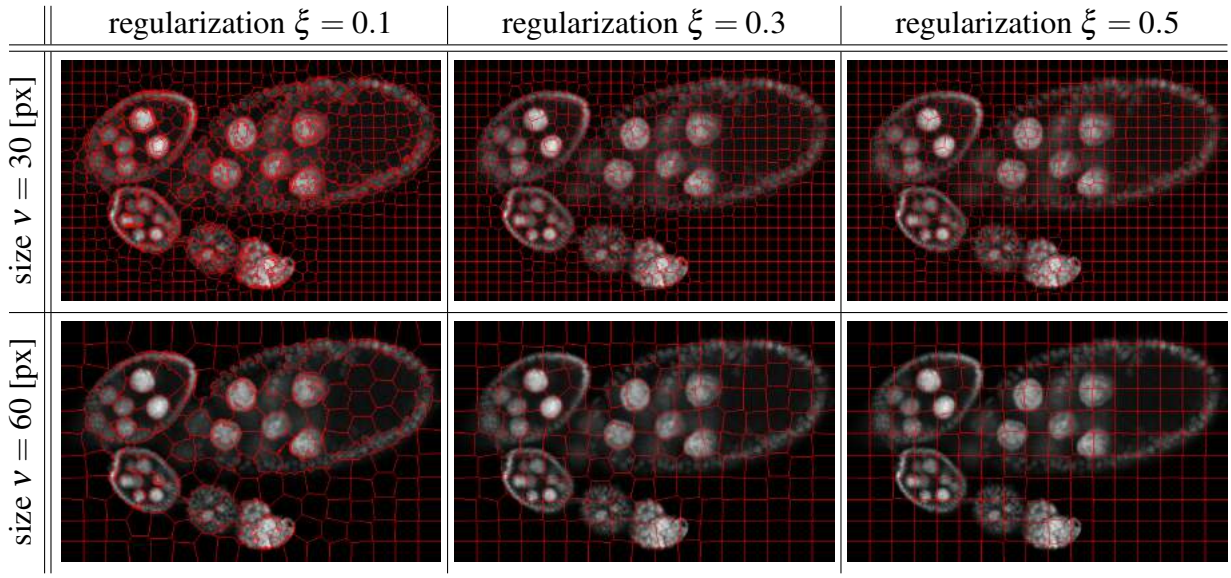


Figure 4.6: The influence of the SLIC parameters on Drosophila ovary: superpixel size and regularization.

4.6 Summary

We have presented enhancements for the SLIC superpixel clustering with better performance than the original SLIC. Moreover, we proposed a different regularization parameter, which influences the compactness of resulting superpixels and proposes a default value $\xi = 0.2$. The new post-processing step gives more reliable superpixels shapes, with no need of decreasing superpixel size.

5

(Un)Supervised superpixel segmentation

In this chapter, we present image segmentation on superpixels which has become recently very popular in many (and not only) medical and biological applications. It is often advantageous to first group pixels into compact, edge-respecting superpixels, because these reduce the size of the segmentation problem and thus the segmentation time by order of magnitudes. Also, features calculated from superpixel regions are more robust than features calculated from fixed pixel neighborhoods. We present general multi-class image segmentation method, as you can see in Figure 5.1, consisting of the following steps: (i) computation of superpixels; (ii) extraction of superpixel-based descriptors; (iii) calculating image-based class probabilities in a supervised or unsupervised manner; and (iv) regularized superpixel classification using Graph Cut. We illustrate particular steps of this segmentation pipeline on real medical imaging, see Section 5.6.1 and 5.6.2.

Later in Section 5.6 we present obtained performance on real-word medical images and compare them with standard methods. We show that unsupervised segmentation provides similar results to the supervised method, but does not require manually annotated training data, which is often expensive to obtain.

This segmentation utilizes superpixels presented in Chapter 4 and the segmented images are used in following Chapter 6 for computing label histogram and determining the object shape, and transformed in the form of a probability map of being a part of an object in Chapter 7.

The text in this chapter is strongly based on the text of the paper [40], an extended version of the paper [73]. Several images were enlarged, and some notations were changed to unify with the rest of the thesis. The part describing superpixels was reduced since it has been presented in Chapter 4.

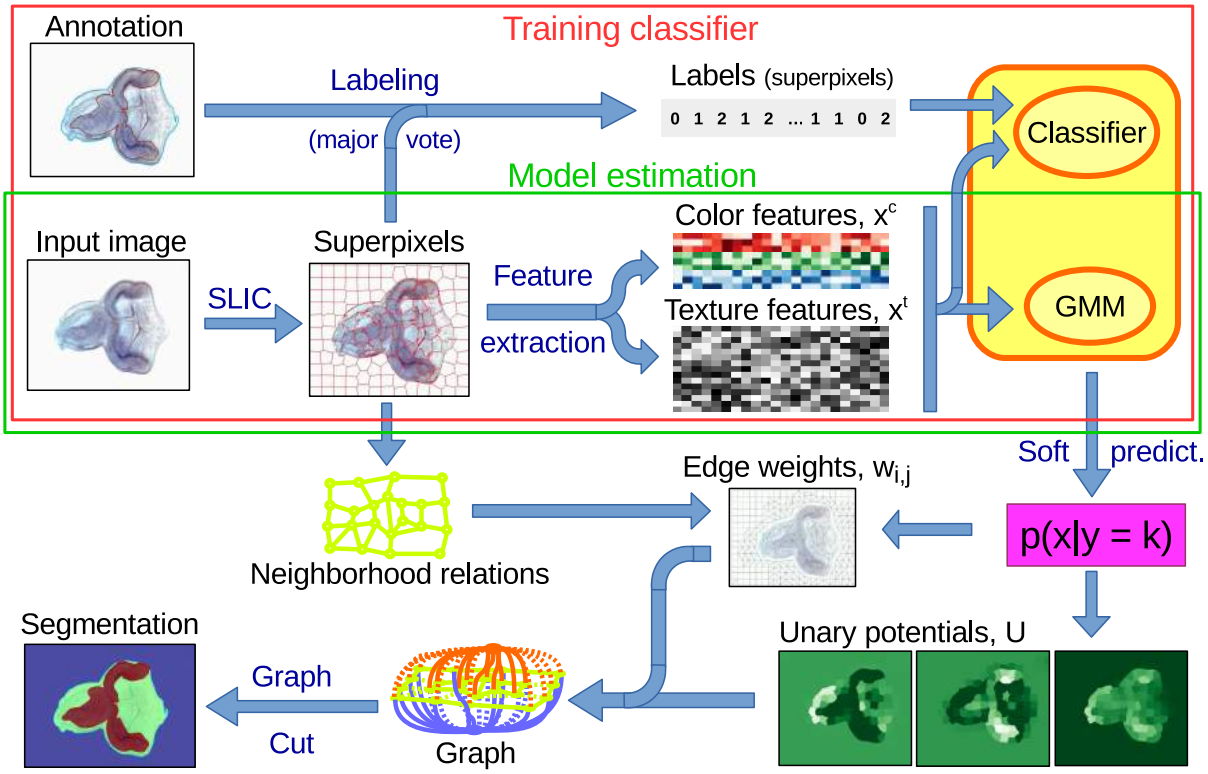


Figure 5.1: The scheme of the complete image segmentation pipeline with supervised and unsupervised learning.

5.1 Task formulation

The input is a color or a gray-scale image, and we want to segment it into a small number of classes which can be trained from a set of training examples or learned in an unsupervised way if those classes are well separable. For the resulting segmentation, we also require a certain level of spatial compactness, i.e., adjacent superpixels with a similar local property should form a part of the same class.

Formally, let us have an input image $I : \Omega \to \mathbb{R}^m$ defined on a pixel grid $\Omega \subseteq \mathbb{Z}^d$. We shall decompose the image pixels into superpixels, $\Omega = \bigcup_{s \in S} \Omega_s$, where Ω_s denotes pixels belonging to a superpixel s and S is a set of all superpixels. We will then try to find a superpixel segmentation function $\mathbf{y} : S \to \mathbb{L}$, where $\mathbb{L}$ is a set of class labels. Superpixel segmentation Y can easily be interpolated to all pixels by assigning its label to all pixels within a superpixel, yielding a pixel-level segmentation function $\mathbf{y}_\Omega : \Omega \to \mathbb{L}$. The scheme of the complete segmentation pipeline is presented in Figure 5.1.

5.2 Minimization problem

For each superpixel $s \in S$, we compute a vector of (color and texture) features $x_s \in X$. We find the superpixel classes $\mathbf{y}_s = Y(s)$ from the maximum a posteriori (MAP) estimate

$$Y^* = \arg\max_Y P(Y|X) = \arg\max_Y \frac{p(X|Y) \cdot P(Y)}{p(X)} \tag{5.1}$$

where $P(Y)$ is the *a priori* probability of a specific segmentation (of all pixels) regardless of the descriptors, and $p(X|Y)$ is the conditional density of the set X of all descriptors x_s given the labels Y, and $p(X)$ is the marginal probability density function of X, which is constant for given image I and independent on Y.

We express the spatial dependency of superpixel labels using a Markov random field and factorize the term $P(Y)$ as follows

$$P(Y) = \prod_{s \in S} h(\mathbf{y}_s) \cdot \prod_{(i,j) \in \mathcal{N} \subseteq S^2} R(\mathbf{y}_i, \mathbf{y}_j) \tag{5.2}$$

The first term, $h : \mathbb{L} \to \mathbb{R}$, is the prior probability of each class, independent of position. The second term $R(\mathbf{y}_i, \mathbf{y}_j)$ describes the relationship between the classes of neighboring superpixels, favoring neighboring superpixel classes to be the same. The R can be learned from reference segmentation or can be designed by the user. Because superpixel features x_s are conditionally independent given Y, equation (5.1) can be written as follows:

$$Y^* = \arg\max_Y \prod_{i \in S} \big(p(x_i|\mathbf{y}_i) \cdot h(\mathbf{y}_i)\big) \cdot \prod_{(i,j) \in \mathcal{N}} R(\mathbf{y}_i, \mathbf{y}_j) \tag{5.3}$$

Applying the logarithm to equation (5.3), we obtain the widely used Potts model which can be solved by Graph Cuts [98], where we minimize the sum of unary and pairwise potentials

$$Y^* = \arg\min_Y \sum_s \underbrace{-\log\left(p(x_s|\mathbf{y}_s) \cdot h(\mathbf{y}_s)\right)}_{U_s(\mathbf{y}_s)} + \sum_{(i,j) \in \mathcal{N}} \underbrace{-\log R(\mathbf{y}_i, \mathbf{y}_j)}_{\beta w_{i,j} B(\mathbf{y}_i, \mathbf{y}_j)} \tag{5.4}$$

where we have factorized the regularization into a global coefficient β, position-dependent weight w_{ij} and a label-dependent part B. The unary term $U_s(\mathbf{y}_s)$ represents the observations (image) and the a priori class probabilities, and the binary potential $B : \mathbb{L}^2 \to \mathbb{R}$ leads to spatial regularization. If training data for estimating R is not available, we set $B(k,l) = [\![k \neq l]\!]$.

5.3 Superpixels and Feature space

We use the Simple Linear Iterative Clustering [87] (SLIC) algorithm to calculate the superpixels so that the superpixels are compact both in space and in color. More details are in Section 4.

We create a feature vector $x_s = \left[x_s^c \; x_s^t\right]$ for each superpixel s, containing color features x_s^c and texture features x_s^t [56]. The features (or descriptors) are normalized element-wise as $\bar{x} = \frac{x - \mu_x}{2(3\sigma_x + 1)}$ where μ and σ are vectors of the means and standard deviations of each component over the whole dataset.

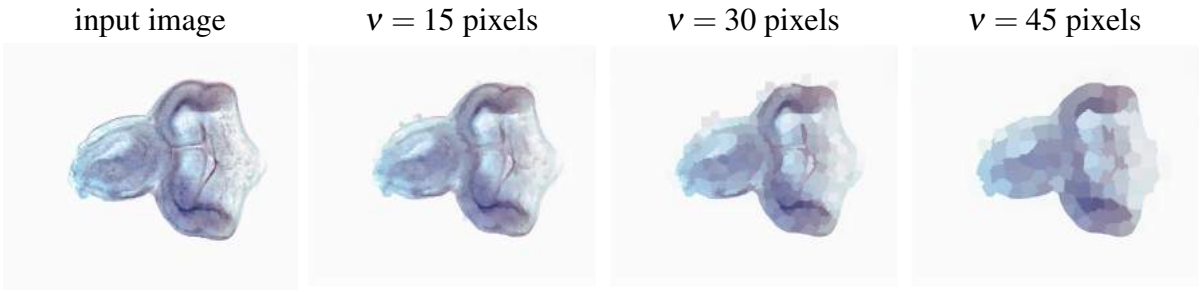


Figure 5.2: We show the input image (a) and its approximation using mean colors of the superpixels. The SLIC superpixel regularization is $\xi = 0.2$, and the sizes are 15, 30 and 45 pixels.

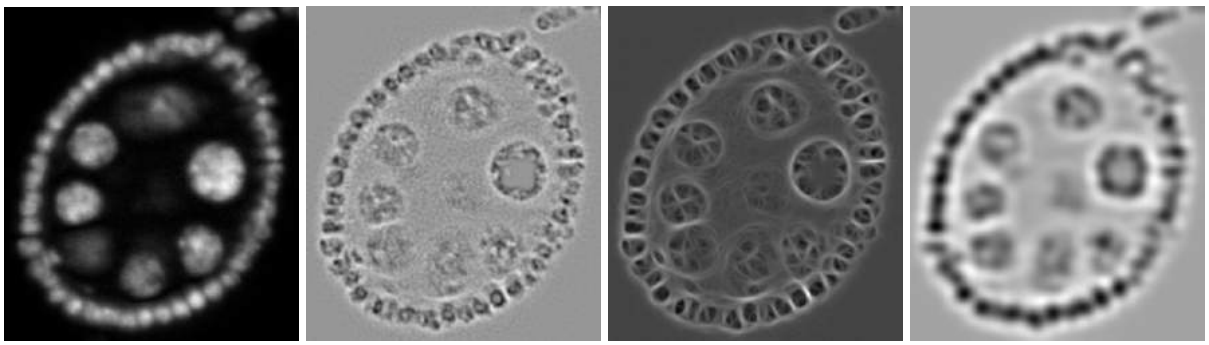


Figure 5.3: Sample gray-scale image (Drosophila ovaries) *(a)* and responses to some Leung-Malik filter banks *(b-d)*.

5.3.1 Color features

For each color channel I^c, we define a feature vector $[\mu_s(I^c), \sigma_s(I^c), E_s(I^c), M_s(I^c)]$, where the components are the mean, the standard deviation, the energy, and the median of that color channel over the superpixel. To get the full color feature vector x_s^c, the vectors for all three-color channels are concatenated. The mean color feature is illustrated in Figure 5.2. Different color spaces or a combination thereof can be used (e.g., RGB, HSV, Lab) [75]. In our experiments, we used RGB color space.

5.3.2 Texture features

The Leung-Malik (LM) [164] filterbank is a multi-scale, multi-orientation filterbank with 48 filters in total. It consists of the first and second derivatives of a Gaussian at 6 orientations; 8 Laplacians of Gaussian filters; and 4 Gaussians. To characterize the texture, we compute the responses $F_j^c = I^c * \mathtt{LM}_j$, for each color channel independently [56] (see Figure 5.3 for examples). To achieve rotation invariance, we take for each pixel the maximum response over all orientations, determining the orientation and thus reducing the number of features per channel to 18. As above, for each filter and each color channel, we calculate the mean, the standard deviation, the

energy, the median, and also the mean gradient amplitude $G_s(F_j^c) = {}^1/_{|\Omega_s|} \sum \left\| \nabla F_s^{c,j} \right\|$. To get the feature vector x_s^t, the vectors $[\mu_s(F_j^c), \sigma_s(F_j^c), E_s(F_j^c), M(F_j^c), G_s(F_j^c)]$ are concatenated for all j and for all color channels c.

5.4 Multi-class modeling

For each class $k \in L$, we define a model $p(x|\mathbf{y}_s = k)$ for the feature vector probability density, with parameters θ_k.

5.4.1 Gaussian mixture model

In the unsupervised case, we use the Gaussian Mixture Model (GMM) [71]. We assume that for each class, the distribution on x_s is normal: $p(x_s) = \sum_{k \in L} \rho_k N(x_s; \mu_k, \Sigma_k)$ for $\mathbf{y}_s = k$. The overall probability $p(x_s)$ for unknown $\mathbf{y}_s$ is therefore a mixture, with mixing probabilities $h(k)$, used in (5.3) and (5.4). Parameters μ_k, Σ_k, and $h(k)$ are estimated using the Expectation-Maximization (EM) algorithm [96]. The basic scenario is an estimation of GMM for each image independently. This can be used in the situation when each segmented image is different from the others. Another case is estimating GMM over a set of images where we expect a similar appearance model. Both approaches are explored in Section 5.5.2.

5.4.2 Classifier-based model

In the supervised case, we train a standard classifier such as Random Forest, logistic regression, k-Nearest Neighbors (KNN), Support Vector Machine (SVM) or Gradient Boost on a set of training examples. All these classifiers can produce a posteriory probability $p(\mathbf{y}_s|x_i)$, which we plug in equation (5.1) instead of $p(x_i|\mathbf{y}_i)h(\mathbf{y}_i)$ which then tuns to be conditional MRF. The class labels are usually provided pixel-wise in the training data. We convert pixel labels to superpixel labels by taking the majority class. Superpixels, where less than 98 % belong to a single class are ignored — this way we lose less than 5% of the training instances while avoiding misleading training data.

5.5 Graph Cut

5.5.1 Potentials

Following (5.4), we set the unary potentials U_s^k. The visualization is presented in Figure 5.4 for an example image where the GMM for 3 classes was learned using the EM algorithm.

From the training data, we can learn the probability $p(k,l)$ In the case of non-availability of labeled training data, we turn to the use of unsupervised learning, e.g., the Gaussian mixture model. We create the $B(k,l)$ matrix uniformly with zeros on the diagonal and with ones elsewhere.

Figure 5.4: Visualization of superpixels *(a)* colored according to the computed unary potential U^k for classes $k = 0, 1, 2$ where the intensity of the green color reflects the potential. The class models were estimated from the image as GMM where images *(b-d)* are background, gene activation, and the non-activated disc, respectively.

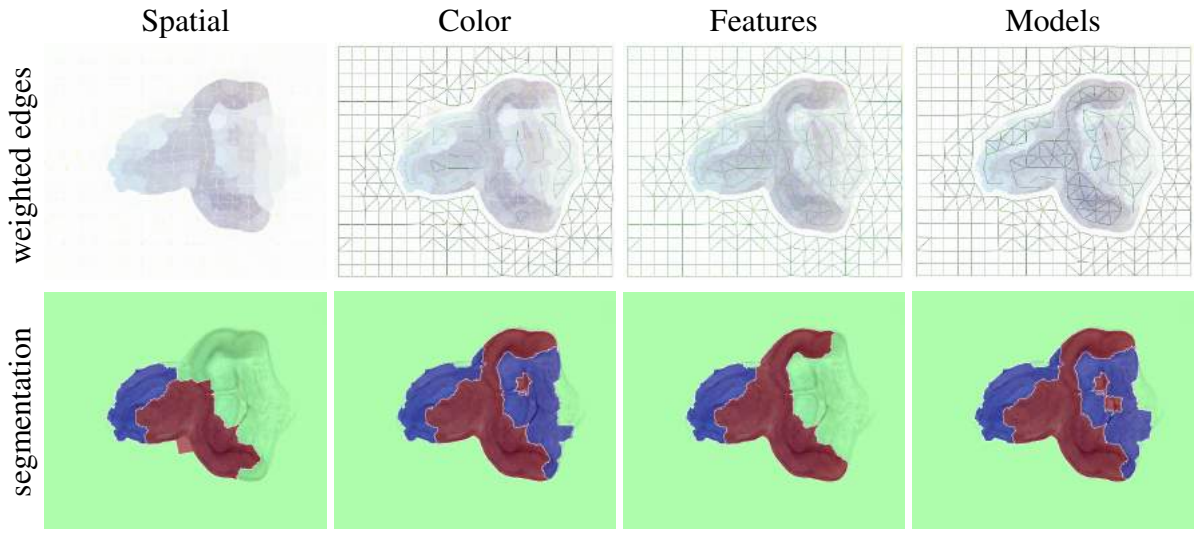


Figure 5.5: The edge weights $w_{i,j}$ represent the intensities of the green lines *(top row)* with the resulting segmentation *(bottom row)*. The same U and B potentials are used. The reference segmentation is presented in Section 3.1.1.

5.5.2 Edge weights

Let us turn our attention to the weights w_{ij} in (5.4), which we shall call edge weights, as they correspond to edges between nodes of the graph in the Graph Cut method. In our case, the nodes are superpixels. The idea is to penalize class boundaries between similar superpixels, and so the potentials can be seen as global pasteurization, and the weighted edges can be seen as local regularization. Taking advantage of the rich superpixel descriptors, we show several formulas for w_{ij}, three from the literature and one new formula. The weights are normalized by the Euclidean distance $d_E(i, j)$ between the two objects [78], in our case superpixel centers $i, j \in S$, to compensate for nonuniformly distributed superpixels. The $\bar{d}_E^S$ is the mean Euclidean distance between all neighboring superpixels $d_E(i, j)$.

Spatial weighting is given by $w_{i,j} = \frac{\bar{d}_E^S}{d_E(i,j)}$. It is the simplest case, and it can be seen as a distance weighted Potts model.

Color weighting Normalized color distance [78, 165] is also often used, assuming that superpixels with similar colors should be grouped. We use the Euclidean distance $d_E(\cdot,\cdot)$ of the superpixel mean colors $\bar{I}_s$ in the RGB color space, normalized by its standard deviation σ_c over all superpixel colors.

$$w_{i,j} = \exp\left(-\frac{d_E(\bar{I}_i,\bar{I}_j)}{2\sigma_c^2}\right) \cdot \frac{\bar{d}_E^S}{d_E(i,j)} \tag{5.5}$$

We have observed that this approach works well for color images. For images without significant color differences, it has similar values as a spatial edge weight above.

Feature weighting Color distance is a special case of a distance between feature vectors. Another option is to take the Manhattan distance $d_M(\cdot,\cdot)$ between the complete feature vectors [102] x_s, normalized by their standard deviation σ_X over all superpixel features. We use Manhattan distance because the order of elements in feature vectors may be arbitrary.

$$w_{i,j} = \exp\left(-\frac{d_M(x_i,x_j)}{2\sigma_X^2}\right) \cdot \frac{\bar{d}_E^S}{d_E(i,j)} \tag{5.6}$$

The disadvantage of feature weighting is that it may give too much weight to irrelevant features.

Model weighting. A new edge weighting that we propose is based on the comparison of a posteriori class Probabilities derived from the learned model. The rationale is that the model 'knows' what is important for a particular application, by translating the high-dimensional feature vector into a low-dimensional vector of $|L|$ probabilities. We compute the l_∞ (Tchebychev) distance $d_T(\cdot,\cdot)$, even other metrics can be used, between the vectors of (non-normalized) class probabilities $p_s^k = p(x_s|\mathbf{y}_s = k) \cdot h(\mathbf{y}_s) \propto P(\mathbf{y}_s = k|x_s)$. The probability differences d_T are normalized by their standard deviation σ_p over all distances.

$$\begin{aligned} d_T(p_i,p_j) &= \max_k \left(|p_i^k - p_j^k|\right) \\ w_{i,j} &= \exp\left(-\frac{d_T(p_i,p_j)}{2\sigma_p^2}\right) \cdot \frac{\bar{d}_E^S}{d_E(i,j)} \end{aligned} \tag{5.7}$$

Figure 5.5 shows examples of the use of different edge weights. The particular selection of the distance metric experiments in Section 5.6.2. The philosophy behind this edge weight is to penalize dissimilarity among neighboring models and to set the minimum weight to very likely equally labeled superpixels. In this sense, we seek for the maximal difference in probability that two neighboring superpixels are assigned to the same class. Figure 5.5 shows that the computed edge weights are very similar to the color distance. This naturally leads to very similar resulting segmentation of the sample image. A further advantage is that the edge weights inside the disc also distinguish the two classes - just the disc and gene activation.

Dataset	image size [px]	color space	SLIC size [px]	SLIC regul.	Features	Classif. / Model	GC regul.
imag. disc	1600	RGB	25	0.25	color & texture	RF (20 trees)	2.
ovary	1000	gray	15	0.35	texture	RF (20 trees)	3.

Table 5.1: Default configuration of the segmentation parameters for each dataset together with the typical image size for each dataset.

Dataset	Random forest	Dec. Tree	Grad. boost	Log. regression	k-NN
imaginal disc	**0.9901**	0.9262	**0.9923**	0.9898	0.9873
Ovary	**0.9902**	0.9627	**0.9913**	0.9884	0.9863

Table 5.2: Evaluation of the performance of the classifiers, as measured by the AUC for the best parameter configuration.

5.6 Experiments

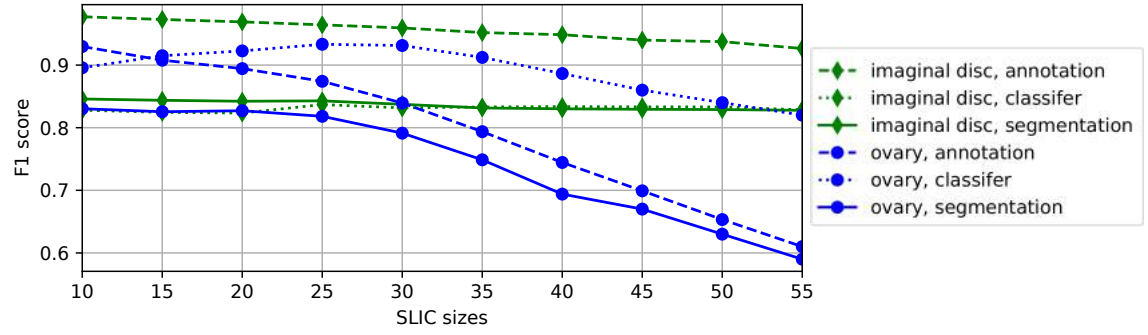
Unless specified otherwise, all experiments use the parameters given in Table 5.1 and we report the F_1 measure using 10-fold cross-validation. We first show the effect of varying the most important parameters and design decisions on the segmentation quality. We then compare the proposed method with previously used methods.

5.6.1 Superpixel parameters

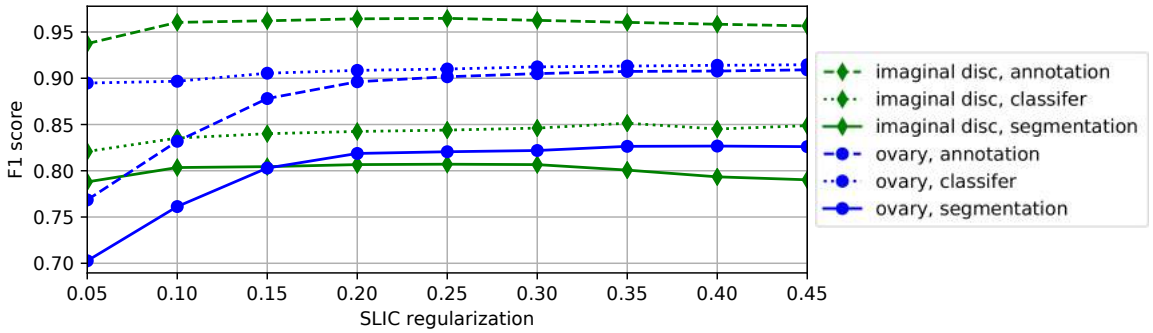
For each dataset, we computed the 'ideal' segmentation Y_A for given superpixels by assigning to superpixels the most frequent class of their pixels in the ground truth. We then evaluated quantitatively the effect of superpixel parameters on the segmentation quality (Figure 5.6) using 10-fold cross-validation. We compute the F_1 measure of the random forest classifier on the superpixel level (denoted 'classifier') and distributed down on the pixel-level Y_Ω (denoted 'segmentation'), with the superpixel-level F_1 measure using the ideal segmentation Y_A (denoted 'annotation'). We see that the optimum varies between applications, but the chosen values of $\nu = 25$ and $\xi = 0.2$ seems to be a good compromise in all cases.

5.6.2 Classifiers and Graph Cut parameters

Classifiers. We experimented with 5 standard classifiers— random forest, decision trees, gradient boost, logistic regression, and k-nearest neighbors, as implemented in the Scikit-learn [166] Python library. For each of them, we chose the best parameters in terms of the F_1 score on superpixels as samples from 250 randomly generated parameter values, sampling uniformly from a user-defined range of values. The classifiers are compared in Table 5.2 using the AUC criterion. We chose random forest because it is one of the best performing classifiers, as it is fast and can handle large datasets.



(a)



(b)

Figure 5.6: Evaluating the F_1 score for different superpixel sizes *(a)* and regularization *(b)* for the random forest on the superpixel level ('classifier') and on the pixel level Y_Ω ('segmentation'), compared with the ideal segmentation Y_A on the superpixel level ('annotation').

Edge weights. We evaluate the effect of choosing different metrics in the model edge weighting (Section 5.5.2) in Table 5.3. The l_∞ metric was chosen as the best compromise. Table 5.4 compares the effect of different edge weight types on the segmentation results. We can see that the newly proposed edge weight (Model weight) based on model distance outperforms all others on all datasets.

Graph Cut regularization. We studied the influence of the Graph-Cut regularization constant β defined by equation (5.4). The optimal value of β depends on the dataset (see Figure 5.7) but the best value in all three cases was within the range $\beta = (1,3)$.

5.6.3 Baseline methods

Together with the baseline methods specific for Drosophila ovaries, we introduce two more general supervised segmentation methods which were used for experimental comparison.

Weka segmentation with Graph Cut We implemented two segmentation methods based on pixel-wise random forest classification [167] using the Weka toolbox [48, 168]. For the dro-

Dataset	nb. classes	l_1	l_2	l_∞
imaginal disc	3	**0.813**	0.808	0.807
ovary	4	0.818	0.816	**0.824**

Table 5.3: The effect of the metrics used in model weights (Section 5.5.2) on the final segmentation accuracy measured by the F_1 score.

Datasets	Spatial weight	Color weight	Features weight	Model weight
imaginal disc	0.6846	0.7985	0.6726	**0.8133**
ovary	0.7379	0.8150	0.7604	**0.8236**

Table 5.4: Comparison of the Graph Cut edge weight types defined in Section 5.5.2 in terms of the F_1 score.

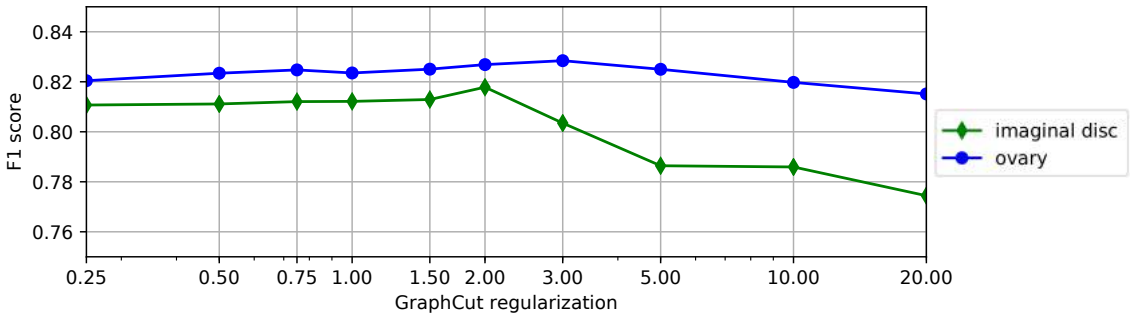


Figure 5.7: Evaluating the F_1 score for different Graph-Cut regularization parameters β.

sophila imaginal discs, we used RGB color channels as features. For the gray-scale drosophila ovary, we used the following texture features calculated from a small neighborhood [168]: mean, median, and a Sobel filter response. The classifier was trained using reference pixel-level segmentation created by an expert. Regularization was applied to the resulting probability maps generated from the classifier by Graph Cut [169] to obtain the final segmentation. The method is denoted as 'Weka & GC(*smoothness cost*, *edge cost*)' in Section 5.6.

Superpixel segmentation. We have also tested the method proposed here composed of only first three steps without Graph Cut regularization, taking the classifier output directly as the final result. It is denoted as "our RF" because it uses the random forest classifier.

5.6.4 Segmentation performance and evaluation

Our key experiment is a comparison of the performance of all methods described here and shown in Table 5.5. Our supervised segmentation is denoted as 'RF', because it uses the random forest classifier. The unsupervised segmentation is denoted as 'GMM', because the model is estimated via the Gaussian mixture model (GMM). A variant of GMM which learns from all

		Method	imaginal disc	ovary
Pixel-wise	Supervised	Weka	0.6923	0.5800
		Weka & GC(0, 100)	0.6887	0.5810
		Weka & GC(1, 50)	0.6887	0.5965
		Weka & GC(10, 50)	0.6887	0.1395
		Weka & GC(50, 100)	0.6850	0.6007
Superpixels		ideal segm. Y_A	0.9696	0.9067
		Supertextons	-	0.7488
		our RF	0.8181	0.8201
		our RF & GC	**0.8229**	**0.8600**
	Unsuper.	our GMM	0.7542	0.5967
		our GMM & GC	*0.7644*	0.6039
		our GMM [gr]	0.7301	0.6009
		our GMM [gr] & GC	0.7564	*0.6083*

Table 5.5: A comparison of all applicable methods for all datasets with ground truth segmentations in terms of the F_1 score. The baseline methods are 'Weka' (a Fiji plugin with Graph Cut regularization, see text) and 'Supertextons' [56] for segmentation the Drosophila ovaries. We also introduce superpixel segmentation Y_A with an ideal classifier.

images at once is denoted by '[gr.]'. Otherwise, the GMM model is learned for each image separately.

Denoted as 'ideal segm. Y_A', it uses the 'ideal' segmentation for given superpixels (Section 5.6.1). It is not a real method, as it needs to know the reference segmentation. It is meant to illustrate how close we are to the performance limit given by the superpixels. However, note that Y_A is optimal in the number of misclassified pixels, i.e. the sum of false positives and false negatives, which may not always translate to the best F_1 score, although the differences are usually small.

It can be seen that, on all datasets, our supervised segmentation with Graph Cut regularization works better than all other methods (except 'ideal segm. Y_A', which is not a real method). The use of Graph Cut regularization improves the segmentation results both for supervised segmentation and for unsupervised segmentation. More detailed discussion follows.

Drosophila imaginal discs

Detailed quantitative results on the Drosophila imaginal disc segmentation are show in Table 5.6. As before, our superpixel segmentation performs better that the pixel-wise baseline 'Weka' method. Moreover, on this dataset, the baseline results are also matched or exceeded by the unsupervised GMM methods. In Figure 5.8, we show an example of segmentation by the supervised and unsupervised methods, together with the reference segmentation. It can be seen that the results are very similar for superpixel segmentations. The three desired classes — gene activation, disc and background — are clearly distinguished.

Speaking about unsupervised segmentation, there is a considerable variance in appearance

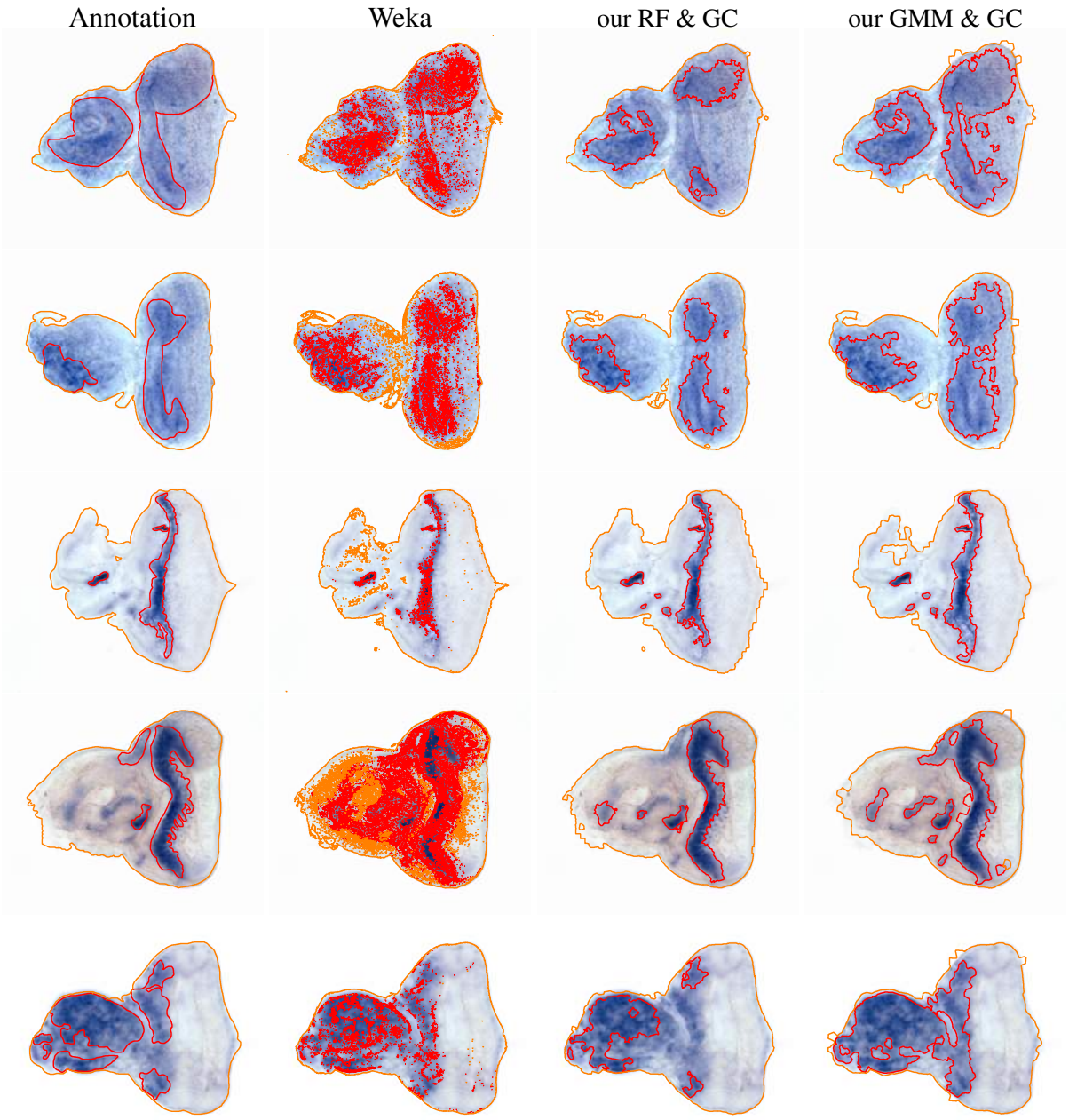


Figure 5.8: Visualization of the expert annotation and segmentation results with the 'Weka' baseline classifier and our supervised and unsupervised methods. Clearly, GMM was learned quite accurately.

		Method	ARS	accuracy	F_1 score	precision	recall
Pixel-wise	Supervised	Weka	0.9508	0.952	0.6923	0.7497	0.7101
		Weka & GC(1, 50)	0.9563	0.9539	0.6887	0.7594	0.7078
		Weka & GC(10, 50)	0.9563	0.9539	0.6887	0.7594	0.7078
		Weka & GC(50, 100)	0.9584	0.9548	0.6850	0.7705	0.7073
Superpixels		ideal segm. Y_A	0.9843	0.9943	0.9696	0.9737	0.9656
		our RF	0.9641	0.9728	0.8181	0.8424	0.8201
		our RF & GC	**0.9653**	**0.9739**	**0.8229**	**0.8506**	**0.8219**
	Unsuper.	our GMM	0.9486	0.9557	0.7542	0.7631	0.8004
		our GMM & GC	0.9504	0.9517	0.7644	0.7844	0.846
		our GMM [gr]	0.9482	0.9533	0.7301	0.7377	0.7803
		our GMM [gr] & GC	0.9521	0.9571	0.7564	0.7643	0.8032

Table 5.6: Segmentation of Drosophila imaginal discs. Quantitative evaluation of applicable methods by standard metrics.

		Method	ARS	accuracy	F_1 score	precision	recall
Pixel-wise	Supervised	Weka	0.8008	0.8842	0.5800	0.6250	0.5833
		Weka & GC(1, 50)	0.8167	0.8909	0.5965	0.6520	0.6023
		Weka & GC(10, 50)	0.8235	0.7844	0.1395	0.1392	0.1399
		Weka & GC(50, 100)	0.8214	0.8934	0.6007	0.6686	0.6085
Superpixels		ideal segm. Y_A	0.9528	0.9735	0.9067	0.9126	0.9021
		Supertextons	0.8633	0.9220	0.7488	0.7403	0.7798
		our RF	0.8836	0.8509	0.8201	0.8298	0.8195
		our RF & GC	**0.8883**	**0.9090**	**0.8600**	**0.8627**	**0.8666**
	Unsuper.	our GMM	0.7306	0.8578	0.5967	0.5854	0.6385
		our GMM & GC	0.7481	0.8649	0.6039	0.5953	0.6472
		our GMM [gr]	0.7385	0.8603	0.6009	0.5831	0.6519
		our GMM [gr] & GC	0.7599	0.8666	0.6083	0.5805	0.6578

Table 5.7: Segmentation of Drosophila ovaries. Quantitative evaluation by standard metrics using baseline pixel-level segmentation, with and without Graph Cut, supertexton [56] segmentation, and the proposed 'RF' superpixel-based segmentation.

which leads to slightly worse results for estimating global appearance model from whole dataset (denoted by '[gr]') than learning the GMM for each image indigently.

Drosophila ovary

Segmentation of Drosophila ovaries turns out to be very challenging, possibly due to the high level of similarity between the two pairs of classes: nurse cells—follicular cells, and cytoplasm—background, which the baseline 'Weka' method fails to distinguish. By contrast, this task is well handled by texture features when superpixels are used.

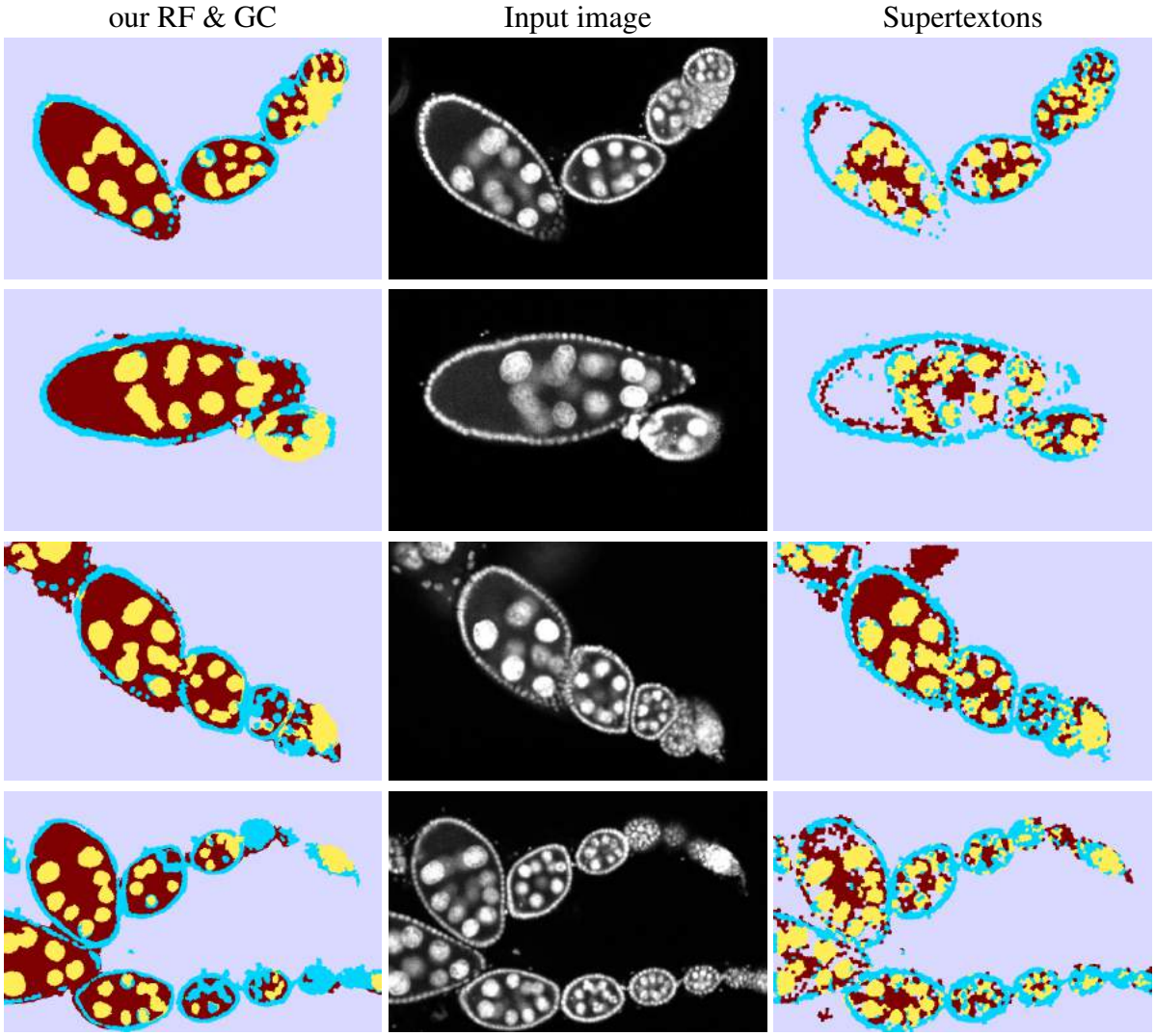


Figure 5.9: Example segmentations of Drosophila ovaries. Observe that our Graph Cut regularized segmentation *(left column)* is much more regular than the baseline method (Supertextons [56]) in right column. The input images are in the middle column.

The quantitative segmentation results are presented in Table 5.7. Our method have much in common with the baseline Supertextons [56]. We improved the F_1 score by 7% by using random forest and texture features instead of the k-NN classifier on the supertextons dictionary. We gained an additional 4% by applying Graph Cut spatial regularization. Some example results are shown in Figure 5.9, and we can observe much better spatial regularity of our Graph Cut regularized method with respect to the baseline 'Supertextons'. The unsupervised methods work less well for this application due to the high level of similarity between segmented classes.

5.7 Summary

We have presented a feature-based segmentation method using an innovative combination of superpixels and Graph Cut regularization to impose spatial compactness. This makes the segmentation both fast and robust. We have also introduced a new edge weight factor. Finally, we have tested our method extensively on five real applications, and have compared it with previously used methods. We found not only that superpixel methods work better than non-superpixel methods, but also that Graph Cut regularization yields further improvement. Also, an advantage is that the supervised segmentation can be trained from just a few images or even partially annotated examples. Interestingly, an unsupervised variant of our method can generate completely acceptable results for some applications, without the need for manual segmentation.

6

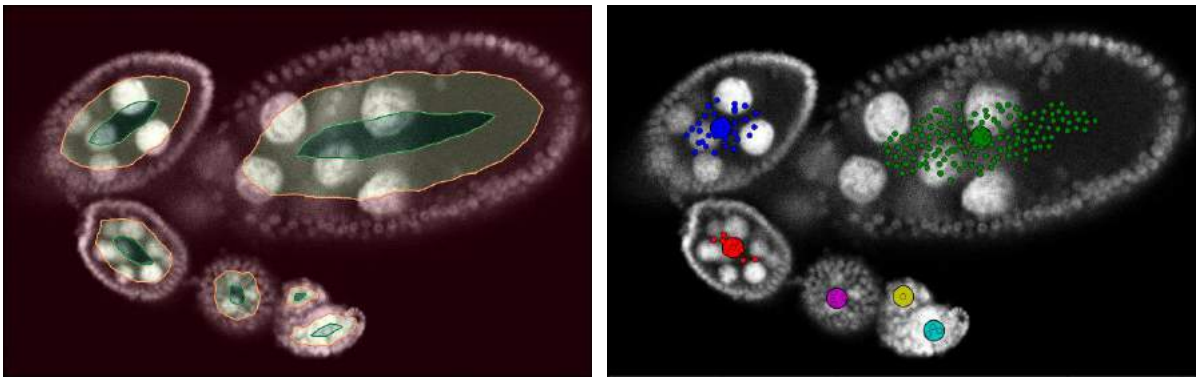
Object center detection and ellipse fitting

This chapter is related to object center detection, namely detection egg centers in microscopy images of Drosophila ovary in structure channel only, see magenta color in Figure 1.2(a). Nevertheless these methods can be used for other tasks where we seek for points with a specific property in high structured segmentations.

The presented image processing pipeline to detect and localize Drosophila egg chambers consists of the following steps: (i) superpixel-based image segmentation into relevant tissue classes presented in Chapter 5. (ii) detection of egg center candidates using label histograms and ray features; (iii) clustering of center candidates and; (iv) area-based maximum likelihood ellipse model fitting. As we present later in Section 6.4, the proposed method can detect most of human-expert annotated egg chambers at important developmental stages with nearly zero false-positive rate, which is adequate for the further analysis. The ellipse approximation of the egg chambers significantly decreases the multiple center proposal which is an issue of a sparse center cloud of center point prediction.

This center detection procedure is based on semantic segmentation presented in Chaper 5 and it utilizes superpixel centers from Chaper 4 as sampled points instead of random sampling. There are two outputs of this pipeline; the extracted centers are used as initial points for region growing in Chapter 7 compare to the object (egg) approximation which serves only to pruning many center detection inside a single egg. Optionally since the ellipses seem to be a good approximation of ovary chamber (in this particular application) this estimate may be used for egg alignment to a typical shape required in BPDL in Chapter 8.

This chapter is strongly based on the paper [38]. Several images were enlarged, and some notations were changed to unify with the rest of the thesis. The part describing superpixel segmentation was reduced since it has been presented in Chapter 5.



(a) (b)

Figure 6.1: *(a)* A center detector is trained using positive central examples in green and negative far away examples in red, ignoring the intermediate zone in yellow. *(b)* Automatically detected central points clustered using DBSCAN. Each cluster is shown in a different color. The centroids of the clusters are drawn as large dots.

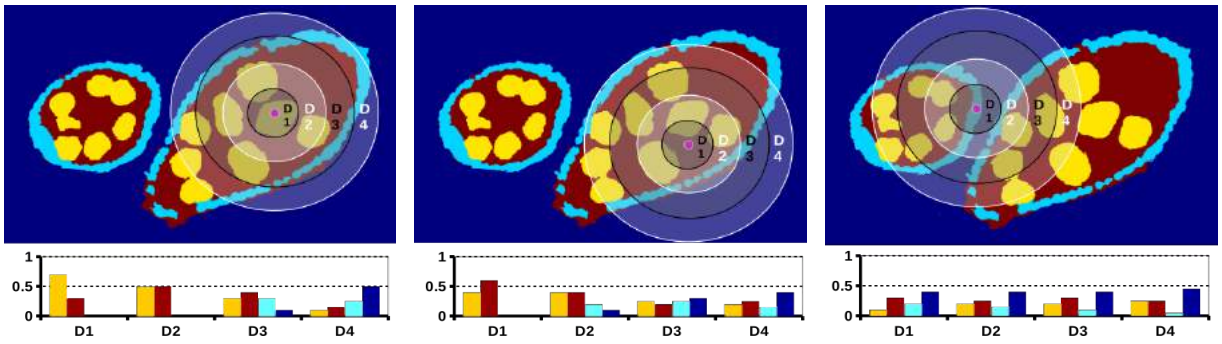
6.1 Methodology

We present an image processing pipeline to detect egg chambers in Drosophila oogenesis. The proposal is divided into four stages: First, we use the methodology proposed in [56, 73] to segment raw ovarioles. Second, a set of features are computed on segmented ovarioles using ray features [136] and a novel labeling technique to detect points within egg chambers. Then, the points detected are clustered to generate ellipses that are the first estimation of the egg chambers. Finally, a maximum likelihood model fitting is applied to the ellipses to adjust the segmentation.

We use supervised superpixel segmentation proposed in Section 5: SLIC superpixels are calculated [87] with an initial size of 15 pixels; for each superpixel, color and texture features are computed; and then, the superpixels are assigned to one of the following four classes (background, follicle cells, cytoplasm, or nurse cells) using a random forest classifier with Graph Cut [98] regularization (see Figure 2.1(b)).

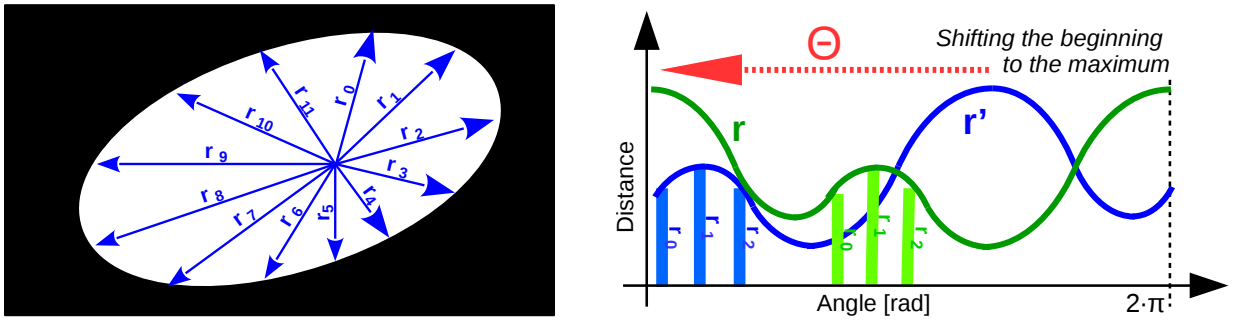
6.2 Center detection

In order to detect points within the central part of the egg chambers, we use two sets of features based on label histograms and modified Ray features [136]. The center candidates are chosen from superpixel centroids using a random forest classifier.The features are normalized to zero mean and unit variance. For training, superpixels close to a center are considered positive (as measured by the relative distance to the background), superpixels far away as negative, ignoring those in-between (see Figure 6.1(a)).



(a) (b) (c)

Figure 6.2: Label histogram descriptor for two different points. A set of annular regions D_i is defined around a reference point, then normalized histograms of label frequencies are computed.



(a) (b)

Figure 6.3: *(a)* A shape is described by ray features, distances from the center to the boundaries in predefined directions. *(b)* The original and shifted distance vectors, $r^{'}$ (in blue) and r (in green) respectively.

6.2.1 Label histograms

The label histograms describe the class distribution in the neighborhood of the point of interest. Here, we are interested in a rotation invariant descriptor; therefore, we used circles, but a more general shape can be used. Around a given point (p_i), a set of N annular regions D_i is defined (see Figure 6.2(a)). For each region, a normalized label (class) histogram is computed, counting the number of pixels of each class within each region is counted. The histograms are concatenated.

6.2.2 Ray features

We use a modified version Ray features [136] were originally proposed in as a shape approximation of convex objects. Here, we assume a binary segmentation $Y^* : P \rightarrow \mathbb{L}^*$ where $\mathbb{L}^* = \{0, 1\}$ with 0: background 1: foreground is union of egg tissues(0-background, 1-follicle cells, 2-cytoplasm, and 3-nurse cells). The Ray features are constructed with a predetermined

angular step ω from a single point of interest (p_i). We measure the distance of each ray $\vec{r}$ to the background boundary and resulting distance vector is $r = \{r_1, \ldots, r_.\}$. We rely on rotation invariance, so we shift the vector r by angle Θ such that it starts with its maxim $r^*_j = r_{j-\Theta}$ (Note the Ray features are periodic so the $|r^*| = |r|$). The orientation invariant Ray feature descriptor for a point p_i is $r = \big(r_1, \ldots, r_n\big)$, with $n = 2\pi/\omega$.

We estimate the boundary points for ellipse fitting as a back reconstruction of Ray features touching the end of boundary cells.

6.2.3 Center clustering

A priory we do not know the number of eggs in each image. On the other hand, the center candidates belonging to an egg are not far each other. Detections corresponding to individual eggs are grouped using density-based spatial clustering (DBSCAN) [170] that handles arbitrarily shaped clusters and naturally detects outliers. The distance threshold parameter of DBSCAN is set to $3\times$ the point sampling distance (superpixel size). Each cluster is represented by its mean position c_k (see Figure 6.1(b)).

6.3 Ellipse fitting and segmentation

We represent each egg by an ellipse (see Figure 6.6). The advantages are: a small number of parameters, convexity, and compactness. The fundamental idea is generating ellipse hypothesis from estimated object (egg) boundary points and chose such ellipse which maximises the assumption that most of the ellipse interior is filled by a class with high probably being an object compares to outside oval being background.

We want to fit an ellipse to each egg estimated by its center, such that, it contains only pixels labelled Y as an egg (0-background, 1-follicle cells, 2-cytoplasm, and 3-nurse cells). In other words, we want to maximize the probability $P(\mathbf{y}|G) = \prod_{i\in\Omega} P(Y_i|G_i)$ that the pixel i is an egg or not, $G : \Omega \to \{0, 1\}$. This can be rewritten as the product of probability being egg in foreground and being background elsewhere

$$\arg\max_{F,B} \prod_{i\in\Omega_F} P_F(Y_i) \cdot \prod_{i\in\Omega\setminus\Omega_F} P_B(\mathbf{y}_i) \tag{6.1}$$

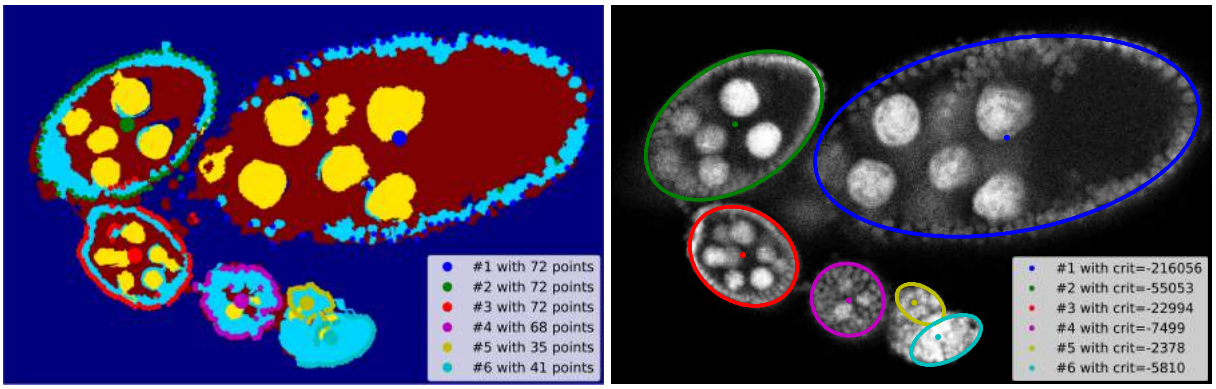
where Ω_F is the ellipse interior and Ω is the entire image. $P_F(Y_i)$ and $P_B(Y_i)$ are the probabilities that a pixel i inside and outside the egg chamber is classified to a class $\mathbf{y}_i$, respectively.

Using negative log likelihood $h_\bullet = -\log P_\bullet$ and substituting

$$\sum_{i\in\Omega} h_B(Y_i) = \sum_{i\in\Omega_F} h_B(Y_i) + \sum_{i\in\Omega\setminus\Omega_F} h_B(Y_i)$$

we obtain an equivalent problem:

$$\arg\min_{F,B} \sum_{i\in\Omega_F} h_F(Y_i) + \sum_{i\in\Omega} h_B(Y_i) - \sum_{i\in\Omega_F} h_B(Y_i) \tag{6.2}$$



(a) (b)

Figure 6.4: *(a)* Ellipse fitting takes as input the four-class segmentation and cluster centers c_k (shown as large dots). Possible ellipse boundary points are found as end-points of rays cast from each cluster center. *(b)* Fitted ellipses are shown overlaid over the segmentation and original image.

where the term $\sum_{i\in\Omega} h_B(Y_i)$ is constant and the minimization problem is simplified:

$$\arg\min_{F,B} \sum_{i\in\Omega_F} h_F(Y_i) - h_B(Y_i) \tag{6.3}$$

Evaluating the criterion inside an ellipse can be accelerated using superpixels. For the four classes of Y we can learn the probability from training examples (accurate pixel-level reference segmentation) or set according to an a priory knowledge, for instance, we have set following $P_B(Y_i = 0) = P_F(Y_i \in \{1,2,3\}) = 0.9$ and $P_F(Y_i = 0) = P_B(Y_i \in \{1,2,3\}) = 0.1$.

Possible ellipse boundary points are determined by casting rays from the center c_k as described in (Section 6.2.2) and taking the first background point along the ray or the first non-follicle class point after a follicle class point (see Figure 6.4(a)). Points on the boundary closer than 5 pixels each other are eliminated to reduce clutter. To obtain a robust fit, we use a random sampling (RANSAC-like) strategy. Ellipses are fitted [171] to randomly selected subsets of 40% of detected boundary points for each center. The best ellipse with respect to Equation (6.3) is chosen as object (egg) approximation.

Note that the initial formulation (6.1) assumes single object in the image, after transformation the criterion (6.3) estimates each object in the image independently. This formulation can be extended for multiple objects with a defined interaction (for instance non-overlapping ellipses) and sharing background (supplement to all objects). Compare to extracting objects independently the joined criterion has a significantly higher number or hypotheses to be evaluated as there are considerably more combinations of object approximations.

Diameters [px]	F_1	AUC	accuracy	precision	recall
[1, 10, 500]	0.5558	0.9033	0.9894	0.4266	0.0756
[10, 25, 500]	0.6465	0.9408	0.9899	0.5202	0.2183
[25, 50, 500]	0.7858	0.9714	0.9925	0.6964	0.5022
[50, 75, 500]	0.8444	**0.9786**	0.9944	0.8142	0.6437
[75, 100, 500]	**0.8836**	0.9757	**0.9951**	0.8422	**0.7037**
[100, 150, 500]	0.8733	0.9731	**0.9952**	**0.8542**	0.6635
[150, 200, 500]	0.7977	0.9392	0.9937	0.8008	0.4703
[200, 250, 500]	0.6769	0.8949	0.9916	0.6439	0.2685
[250, 300, 500]	0.5977	0.8528	0.9911	0.4382	0.1703
[300, 400, 500]	0.5525	0.8146	0.9898	0.4753	0.0648

Table 6.1: We document the influence of triple diameter sizes of label histogram going from small to large to the quality of classification. It is clear that only small or large diameters do not lead to useful features, the best results obtained by including also middle size diameters.

6.4 Experiments

In this section, we present the experimental result on real Drosophila ovary detection from a subset of about four thousand slices where are all stages are approximately equally presented. Just a reminder, the used expert annotation is not exact contour for each egg but approximate bounding box where expert denoted beginning, end and height (at the widest point along main diagonal) of each egg. We present performances from both steps: center detection and positive impact if egg approximation by an ellipse for elimination multiple center reduction.

6.4.1 Center detection performance

In the first group of experiments, we have studied the accuracy of center detection (Section 6.2), formulated as a binary classification problem on superpixels. The area under the ROC curve (AUC) and the F_1 measure were evaluated.

The first observation is that the quality of the initial four-class segmentation is very important. The original segmentation algorithm [56] leads to $F_1 = 0.862$. Using a random forest classifier and Graph Cut regularization, the performance was improved to $F_1 = 0.916$, see Section 5.

When evaluating the influence of the diameters of the annular regions for the label histogram descriptors (Section 6.2.1), we have discovered that it is important to include both small and large regions. With five regions spanning diameters in range from 10 to 300 pixels, we get $F_1 = 0.916$ and AUC $= 0.988$ Including more regions does not significantly improve the results – with 9 regions, we get $F_1 = 0.923$ and AUC $= 0.989$. Those are presented in Table 6.1 and 6.2.

Using a four-class initial segmentation is helpful, with a binary segmentation the performance drops to $F_1 = 0.868$ and AUC $= 0.959$, see Tab 6.3. Concerning ray features (Section 6.2.2), the best performance is obtained for an angular step of $5^\circ \sim 15^\circ$, see Table 6.4. Larger angular steps lead to a loss of details, smaller angular steps increase the descriptor variability. An experiment

Diameters [px]	hard labelling		soft labelling	
	F_1	AUC	F_1	AUC
[5, 10, 20, 30, 50]	0.7308	0.9426	0.7108	0.9609
[10, 25, 50, 75, 100]	0.8861	0.9853	0.8916	0.9836
[50, 100, 150, 200, 250]	0.9147	0.9880	0.9208	0.9847
[200, 250, 300, 400, 500]	0.6737	0.9012	0.6885	0.9119
[10, 50, 100, 200, 300]	0.9164	0.9880	0.9125	0.9885
[10, 25, 50, 75, 100, 150, 200, 250, 300]	**0.9225**	**0.9894**	**0.9282**	**0.9860**

Table 6.2: We can have 4-class segmentation as hard labelling (each pixel has only one label) or soft labelling (each pixel has probabilities of belonging to each class). Again we compare influence of label histogram of small-medium-large diameters as well as dense-sparce diameter sizes. We conclude that having soft labelling does not give any benefit as well as using too many neighborhood diameter sizes.

Merge classes	*none*	{2,3}	{1,2,3}	{0,1} & {2,3}
F_1	**0.9205**	0.9073	0.8898	0.8683
AUC	**0.9830**	0.9821	0.9717	0.9586

Table 6.3: Comparison of classification performances on computed label histogram from 'simplified' segmentation — reduce the number of classes by merging some of them. The merged classes are denoted in brackets {}, for example, {1,2,3} express binary segmentation with background and foreground composed of classes {1,2,3}. We found that decreasing the number of classes in the initial segmentation also has a negative impact on the center classification.

Discrete step ω [deg]	raw		smooth	
	F_1	AUC	F_1	AUC
2	0.8796	0.9726	0.8779	0.9718
5	0.8823	0.9748	0.8750	0.9743
10	0.8851	0.9753	0.8710	0.9712
15	**0.8907**	**0.9746**	0.8648	0.9734
20	0.8851	0.9720	0.8577	0.9640
30	0.8807	0.9742	0.8147	0.9587

Table 6.4: Comparison of angular steps for ray features in combination with optional Gaussian filtering of the r^* vector with σ equal to one discrete step. The observation shows that too small step brings noise to the measurement and on the other hand smoothing vector with large angular steps ω suppresses desired details.

presented in Table 6.5 documents computing ray features to particular a class and assuming up and down edge detecting beginning or ending object, respectively. For example, detecting object boundary as it is shown in Figure 6.3 is detecting down the edge of foreground (in another word up edge of background).

Ray boundary	*'up'*, {0}	*'up'*, {0, 1}	*'down'*, {1}	*'up'*, {0} & *'down'*, {1}	*'up'*, {0} | *'down'*, {1}
F_1	**0.8906**	0.8138	0.7939	0.8796	0.8345
AUC	**0.9770**	0.9389	0.9315	**0.9773**	0.9214

Table 6.5: Evaluation using Ray features touching particular class boundaries. We assume binary segmentation $\mathbb{L}^*$ with 0 - background and 1 - foreground, and two edge types — up ($0 \to 1$) and down ($1 \to 0$). Similar as in Table 6.3, the union of classes being foreground are listed in {} and the rest being background. For example, (*'up'*, {0, 1}) stands for detecting up edge of union of classes 0 and 1 — detecting egg interior without boundary cells, and (*'down'*, {1}) stops at down edge of class 1 — aims to ending of boundaries.

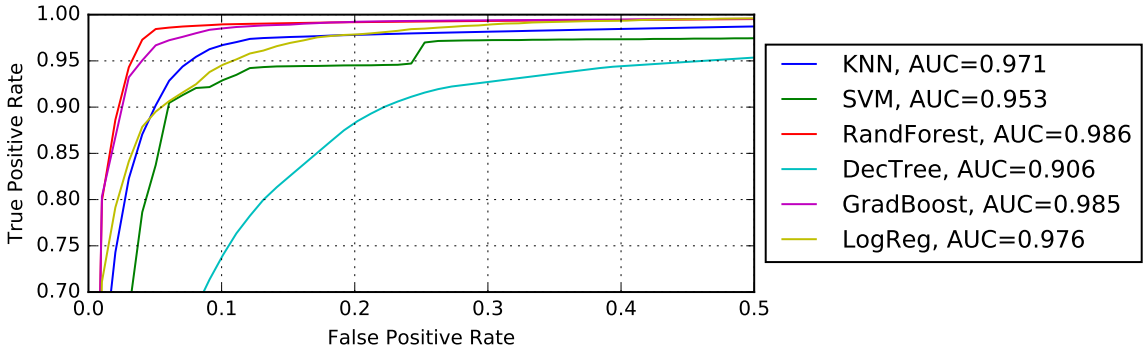


Figure 6.5: ROC curves for different classifiers for the center candidate detection task.

Using both ray features and label histograms is better, yelding AUC $= 0.987$ and $F_1 = 0.930$, than using them separately, with AUC $= 0.986$ and $F_1 = 0.928$ for the label histogram only and AUC $= 0.972$ and $F_1 = 0.884$ for the ray features only.

Finally, we show the ROC curve for several different classifier algorithms (Figure 6.5). We have chosen the random forest classifier which is among the best performing methods and is fast at the same time.

6.4.2 Egg chamber detection

The second part of the experiments evaluates, how many eggs are detected and how many detections are really eggs. The experts marked a subset of 4853 egg chambers with three boundary points (as described above) and a stage label. The results are shown in Table 6.6. The performance on the smallest stage 1 egg chambers is not important for our purposes. Stage 2 is the most challenging. For the rest, less than 1 % of eggs are missed. There is also less than 1 % of false positives, which were counted manually for all stages combined, as not all eggs are marked in the ground truth and the stage information for these object is obviously not available. The most frequent mistake is to detect one egg chamber twice, which can be easily corrected by post-processing — if two ellipses (see Section 6.3) overlap more than 50 % of pixels, we

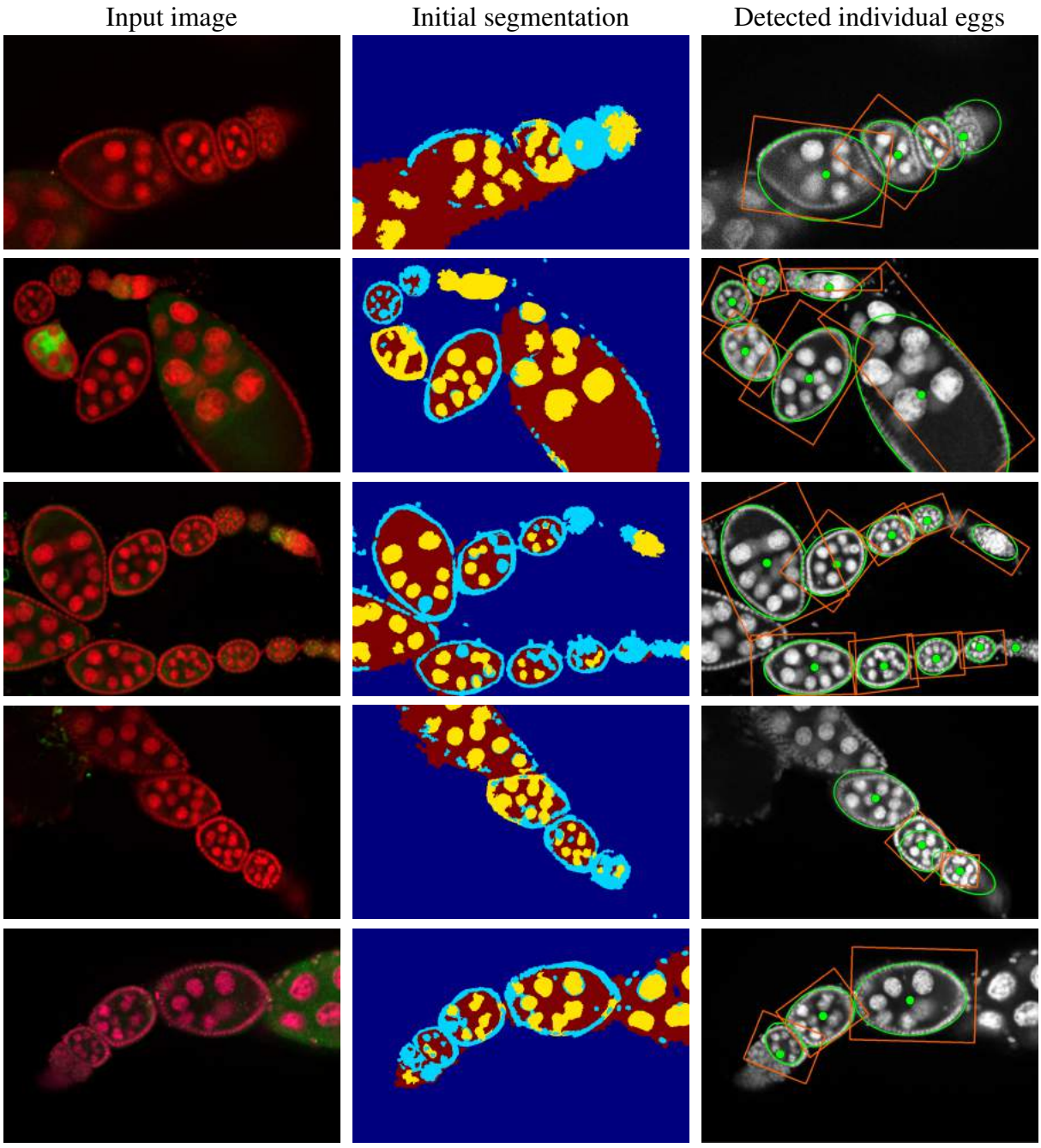


Figure 6.6: Input images *(left)*, initial segmentation *(middle)* followed by the detected centers (cluster means) as dots and the fitted ellipses in green *(right)*. Expert drawn bounding boxes are shown as red rectangles (not all eggs are annotated).

Egg chambers	Stage				
	1	2	3	4	5
number	921	1403	865	834	836
false negatives	306 (33%)	158 (11%)	6 (0.7%)	1 (0.1%)	0 (0.0%)
multiple detections (MD)	37 (4.0%)	31 (2.2%)	109 (12%)	80 (9.6%)	90 (11%)
MD after ellipse fitting	18 (2.0%)	13 (0.9%)	27 (3.1%)	20 (2.4%)	30 (3.6%)
false positives	43 (0.9%)				

Table 6.6: Egg detection performance of the egg detection task by development stages, in terms of false positives, false negatives, and the number of multiply detected eggs before and after post-processing with ellipse fitting.

keep only the larger one. We show the number of multiple detections both before and after this post-processing.

We also evaluate the performance of ellipse approximation on a pixel-level annotated subset of 72 images containing about 200 egg chambers. With respect to a watershed segmentation [57] using the distance from the background class as a feature, we improved the mean ARS from 0.755 to 0.857 and the mean Jaccard index from 0.571 to 0.674.

Figure 6.6 shows examples of successful results, including a few corrected multiple detections and a few undetected egg chambers. Note that the user annotation is neither complete nor accurate, which makes the evaluation challenging.

6.5 Summary

We presented a image-processing pipeline of Drosophila egg chamber detection and localization by ellipse fitting in microscopic images. Our contributions include novel label histogram features, the rotation invariant ray features, and area-based maximum likelihood ellipse fitting. The performance is completely adequate for the desired application — it is important that the number of false positives is small but false negatives are not a problem, as long as a sufficiently high number of egg chambers is detected. In the future, a specialized model could be created for the earliest developmental stages to reduce the number of misses.

7 Region Growing with Shape prior

Region growing is a classical image segmentation method based on hierarchical region aggregation using local similarity rules. In this chapter, we introduce novel region growing method on superpixels which differs from standard region growing in three essential aspects. First, it is performed on superpixels instead of pixels, which brings significant speedup and increases compactness. Second, we use ray features to describe the object boundary and consecutively our method uses learned statistical shape properties that encourage plausible shapes. Third, our formulation ensures that the segmented objects do not overlap and it can segment multiple objects.

The problem is represented as an energy minimization and is solved either greedily, or iteratively using Graph Cuts. We also show the impact of binary and multiclass optimization and consecutively allowing swapping label among touching objects. Experimental comparison with standard methods on real-world biological images are presented in Section 7.3.

This method utilizes the superpixels from Chapter 4 as the primary building block. The segmentation presented in Chaper 5 is converted to an appearance model (a probability map being an object). Finally, the extracted centers from Chaper 6 serves as an initil seed for the region growing.

This chapter is strongly based on the paper [39]. Several images were enlarged, and some notations were changed to unify with the rest of the thesis. The part describing superpixels was reduced since it has been presented in Chapter 4 and Ray features in Section 6.2.2.

7.1 Methodology

Let us start with formal definition. Given an input image containing multiple non-overlapping but possibly touching objects, a seed point for each object, and a shape and appearance model, we shall segment these objects as follows: We group pixels into superpixels S (Section 4) and

for each of them calculate the appearance-based object probability. The regions corresponding to objects are then grown (Section 7.2) using the appearance (Section 7.1.1) and shape (Section 7.1.2) models. The final segmentation is represented by a function $\mathbf{y} : S \to \{0, 1, \ldots, K\}$, which assigns each superpixel $s \in S$ to one of the objects (if $\mathbf{y}(s) \neq 0$) or to the background (if $\mathbf{y}(s) = 0$).

7.1.1 Appearance model

For each superpixel $s \in S$ we calculate a descriptor g_s which represents the appearance of s through its texture or color properties. Given g_s, we use the appearance model to calculate the probability $P_g(g_s)$ that a superpixel s belongs to an object. For notational convenience, we shall write

$$P_g\big(\mathbf{y}(s)|g_s\big) = \begin{cases} P_g(g_s) & \text{for } \mathbf{y}(s) \neq 0 \\ 1 - P_g(g_s) & \text{for } \mathbf{y}(s) = 0 \end{cases} \tag{7.1}$$

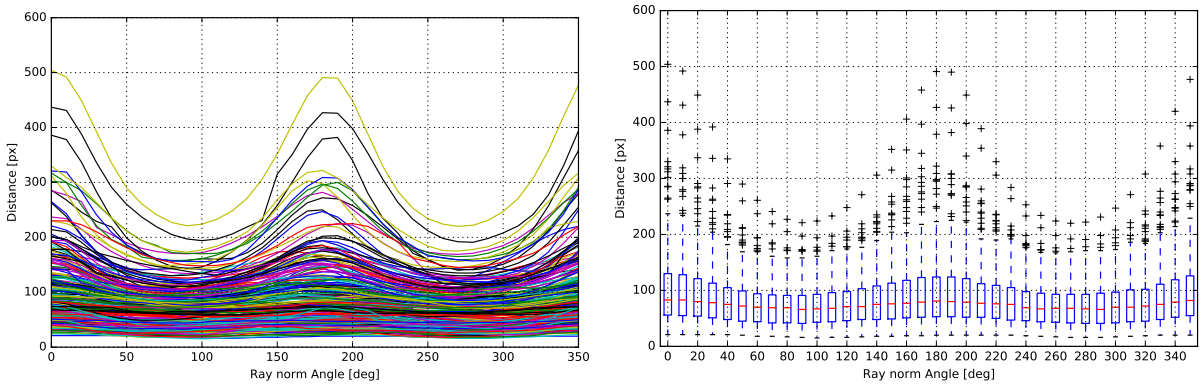
For our application, we take advantage of the fact that we already have a good preliminary segmentation method that can assign superpixels into four biologically meaningful classes (cytoplasm, follicle cells, nurse cells and background) based on texture and color features, and a random forest classifier with Graph Cuts regularization [38, 56]. Our descriptor g_s is therefore simply an integer $\{1, \ldots, 4\}$, representing one of the four classes. The probability $P_g(g_s)$ of a superpixel belonging to an egg given the preliminary segmentation can be estimated from labeled training data. See Figure 2.1(b) for an example of the preliminary segmentation, and Figure 7.4(a) for an example of the probability map P_g.

7.1.2 Shape model

The purpose of the shape model is to determine the likelihood of a particular shape being the desired object (in our case, an egg). Given a region, we calculate its center of gravity c and the so-called ray features [72, 136] r', the distances from c to the region boundary in a set of N predefined directions (see Figure 6.3). To ensure rotation invariance, the distance vector $r' = \{r_0, \ldots, r_{N-1}\}$ is circularly shifted to obtain a rotational normalized vector $r(i) = r'\big((i - \Theta) \bmod N\big)$ such that it starts with the maximum element, $r(0) = \max_i r'(i)$.

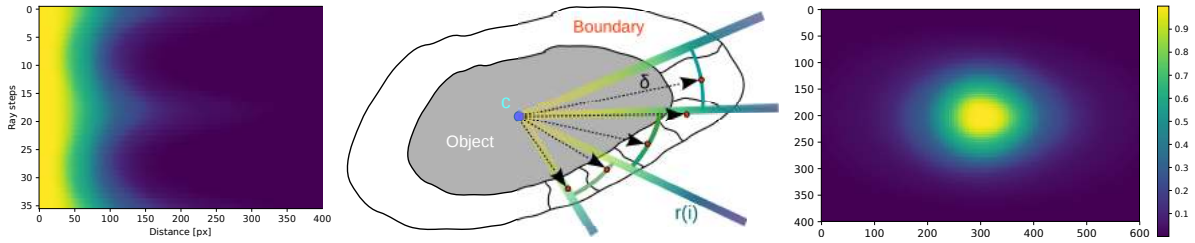
As an example, the ray feature vectors r with $N = 36$ (see Figure 7.1(a)) and the whisker plots for each $r(i)$ independently are shown in Figure 7.1(b) for a set of 250 Drosophila eggs. Invariance to scale can be achieved by another normalization but it is not suitable for our application.

We have chosen to describe the probability density of the ray distance vector r by a simple Gaussian Mixture Model (GMM) with M components over all vectors r assuming diagonal



(a) (b)

Figure 7.1: *(a)* Visualization of 250 egg shapes represented by the distance vectors r and *(b)* their element-wise box and whisker plots for each $r(i)$ with angular step 10^o.



(a) (b) (c)

Figure 7.2: *(a)* Statistical shape model represented as the inverted cumulative probability of ray distance distributions in polar coordinates $i \in N$ and $\delta < r(i)$; *(b)* its mapping (and interpolation) to the Euclidean space using superpixels; and *(c)* the resulting spatial prior q for a single object with zero orientation $\Theta = 0$.

covarinace matrix for each Gaussian

$$p_r(r) = \rho(r) = \sum_{j=1}^{M} w_j f_j(r) \tag{7.2}$$

$$\text{with} \quad f_j(r(i)) = \frac{1}{\sigma_{i,j}\sqrt{2\pi}} \exp\left(-\frac{(r(i)-\mu_{i,j})^2}{2\sigma_{i,j}^2}\right) \quad \text{and} \quad \sum_{j}^{M} w_j = 1$$

The GMM components may represent different egg development stages or significant shape variations. There are $2NM$ model parameters $(\mu_{i,j}, \sigma_{i,j})$ to be estimated from the training data with the Expectation-Maximization (EM) algorithm [96], while the M weights w are estimated for each object independently.

7.1.3 Shape prior

During region growing, we need to calculate the shape prior $P_m\big(\mathbf{y}_s = k|m_k\big) = q(s,m_k)$ that a given superpixel $s \in S$ belongs to an object k, where $m_k = \big[c,r,\Theta,w\big]$ is the shape parameter vector described below. We first calculate the center of gravity c of the region, then calculate the ray features to obtain the shifted distance vector r and orientation angle Θ. Finally, the GMM weights w are obtained by maximum-likelihood fitting of r to the model (7.2).

The particularity of using shape models in the region growing framework is that the shape model needs to allow also intermediate shapes, i.e., shapes that can be grown into likely objects. In other words, the shape described by m_k is not necessarily the shape of the object to be segmented but it may be smaller. Let us denote δ the distance of s from the center of gravity c and let $r = r(i)$ be the corresponding ray along the line from c to s. As $\rho(r)$ from (7.2) is the density of the boundary being at distance r, we see that $q(s,m_k)$ is the cumulative probability of finding the boundary at a distance $\delta < r$, which leads to

$$q(s,m_k) = \int_{\delta}^{\infty} \rho(r)\,\mathrm{d}r = 1 - \int_{0}^{\delta} \rho(r)\,\mathrm{d}r$$

which is easy to evaluate using the cumulative probability density of the GMM. For this calculation, superpixels are represented by their centers. The parameters $\mu_{i,j}$, $\sigma_{i,j}$ are interpolated from neighboring rays using linear interpolation in angles (see Figure 7.2).

The background probability (for $k = 0$) can be calculated as a complement. Given the estimated parameters of all regions $M = (m_1,\ldots,m_K)$, we obtain

$$P_m\big(\mathbf{y}_s = k \mid M\big) = \begin{cases} q(s,m_k) & \text{for } k > 0 \\ \prod_l \big(1 - q(s,m_l)\big) & \text{for } k = 0 \end{cases} \tag{7.3}$$

An example of shape priors for the $M = 5$ GMM components is shown in Figure 7.3.

7.1.4 Variational formulation

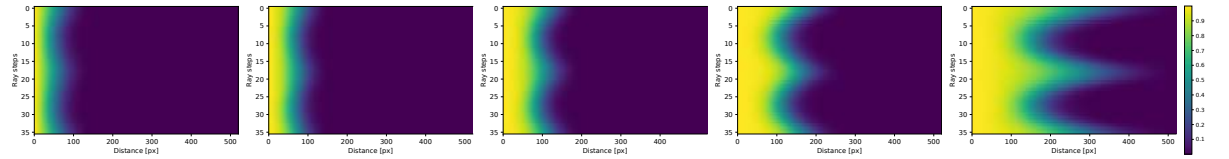
The optimal segmentation $\mathbf{y}^*$ is found by maximizing the a posteriori probability $P(g \mid y,M)$, where g represents the descriptors of all superpixels. We assume that it can be factorized into appearance, shape, and regularization terms as follows

$$P\big(\mathbf{y}_s \mid y,M\big) = \frac{1}{Z(M,y)} P_g(g \mid y) P_m(g \mid M) P_R(g) \tag{7.4}$$

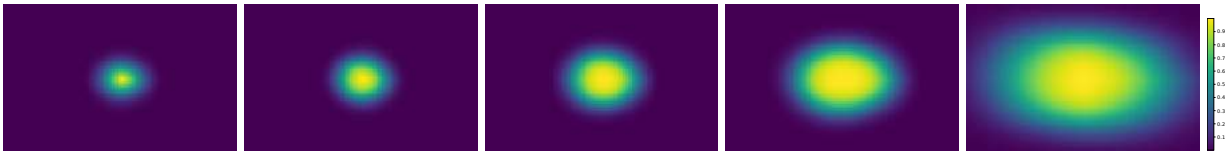
where Z is the normalization factor. The appearance and shape terms P_g and P_m, respectively, are expanded assuming independent pixels as follows:

$$P_g(g \mid y) = \prod_{i \in \Omega} P_g\Big(g\big(s(i)\big) \mid y\big(s(i)\big)\Big) = \prod_{s \in S} P_g\big(\mathbf{y}_s \mid y(s)\big)^{|\Omega_s|} \tag{7.5}$$

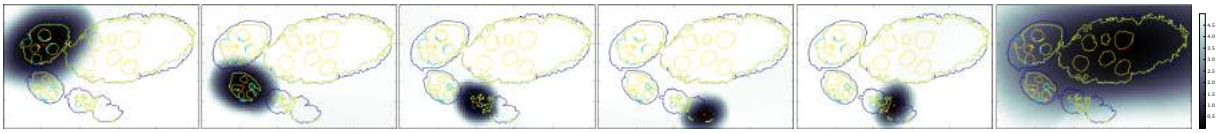
$$P_m(g \mid M) = \prod_{i \in \Omega} P_m\Big(g\big(s(i)\big) \mid M\Big) = \prod_{s \in S} P_m\Big(\mathbf{y}_s \mid M\Big)^{|\Omega_s|} \tag{7.6}$$



(a)

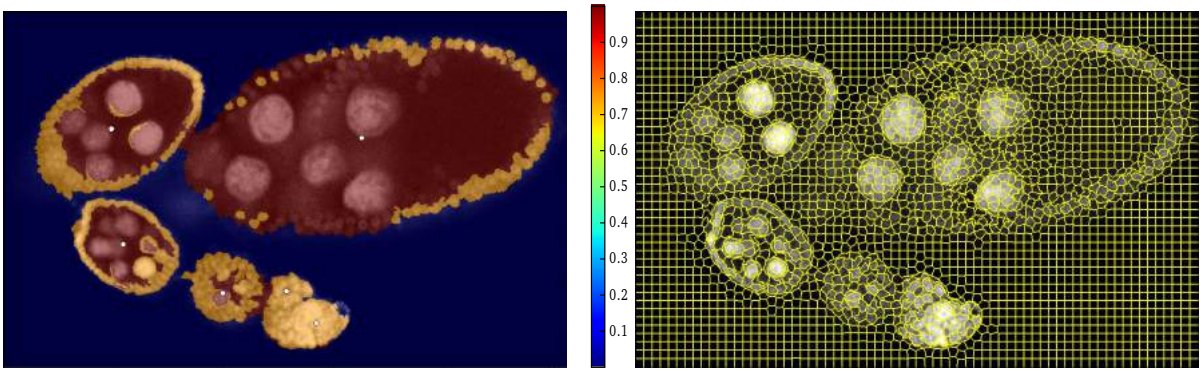


(b)



(c)

Figure 7.3: *(a)* inverted cumulative probabilities of ray distances for $M = 5$ components of the GMM; *(b)* the spatial shape prior q corresponding to each component, and *(c)* shape cost of fitted models to each of the segmented object with thin contours presenting levels of appearance probability P_g.



(a) (b)

Figure 7.4: Sample Drosophila ovary image with multiple eggs. *(a)* Probability map obtained from the preliminary segmentation shown in Figure 2.1(b) representing the likelihood for each superpixels being an egg; *(b)* SLIC superpixels.

where Ω_s are pixels belonging to a superpixel s and $|\Omega_s|$ is the superpixel size. The neighborhood regularization prior P_R is assumed to factorize as

$$P_R(g) = \prod_{(u,v)\in\mathcal{N}_S} H\big(\mathbf{y}(u),\mathbf{y}(v)\big) \tag{7.7}$$

where the product is over neighboring superpixels (u,v) and H is chosen such that it encourages them to belong to the same class, see below.

Taking the negative log-likelihood leads to energy minimization $\mathbf{y}^* = \arg\min_{\mathbf{y}} E(g)$ with

$$E(g) = \sum_{s \in S} |\Omega_s| \cdot \Big[D_s\big(\mathbf{y}_s\big) + V_s\big(\mathbf{y}_s\big) \Big] + \sum_{(u,v) \in \mathcal{N}_S} B\big(\mathbf{y}(u), \mathbf{y}(v)\big) \tag{7.8}$$

where $\mathcal{N}_S$ is set of all neighboring superpixels along the object boundaries, $D_s(k) = -\log P_g\big(k \mid y(s)\big)$ is the data term (described in Section 7.1.1), $V_s(k) = -\log P_m\big(k \mid M\big)$ is the shape term (described in Section 7.1.2).

It remains to define the neighborhood term $B(k,l) = -\log H(k,l)$. The matrix B can be learned from labeled training data. To simplify the task, we shall impose it the following structure

$$B(k,l) = \begin{cases} \omega_0 & \text{for } k = l \\ \omega_1 & \text{for } \min(k,l) = 0,\ k \neq l \\ \omega_2 & \text{otherwise} \end{cases} \tag{7.9}$$

where ω_1 and ω_2 represent penalties for an object superpixel touching a background or another object, respectively; ω_0 can be calculated from the partitioning of unity condition $\sum_{k,l} H(k,l) = 1$. In our case, we obtain approximately $\omega_1 = -\log(0.1)$, $\omega_2 = -\log(0.03)$.

To compensate for model imperfections, it turns out to be useful to add multiplicative coefficients β and γ to modify the relative strength of the three terms:

$$E^{'}(g) = \sum_{s \in S} \underbrace{|\Omega_s| \Big[D_s\big(\mathbf{y}_s\big) + \beta V_s\big(\mathbf{y}_s\big) \Big]}_{U_s(y_s)} + \gamma \sum_{(u,v) \in \mathcal{N}_S} B\big(\mathbf{y}(u), \mathbf{y}(v)\big) \tag{7.10}$$

It can be solved by a standard Graph Cut method [98].

7.2 Region growing

We use an iterative approach to find a labeling $\mathbf{y}$ minimizing the global energy E in (7.10). We alternate two steps: (1) update the shape parameters M for fixed labels $\mathbf{y}$ and (2) optimizing the labels $\mathbf{y}$ for M fixed (see Algorithm 1). The initial object labeling $\mathbf{y}$ is derived from user provided initial object centers c_k. The objects start as small as possible (one superpixel) and grow. For our application, object centers can be obtained automatically using a random forest classifier, neighborhood label histograms, ray features, and density-based spatial clustering. [38]

Updating M is straightforward and quick — for all superpixels S_k currently assigned to object k, we calculate their center of gravity c, the ray distances r, the angle Θ of the longest ray, and the weights w as described in Section 7.1.3.

Let us now consider how to update the superpixel labels $\mathbf{y}$. For speed-up, simplification and in the spirit of region growing, we only allow to change a label of superpixels ∂S_k neighboring an object S_k from the previous iteration and only to a label k. This has the important property that the objects remain compact (simply connected), see Figure 7.5. We have considered four optimization strategies for the superpixel labels.

Algorithm 1: Region growing.

Input: S: superpixels, g: superpixel descriptors, c_k: initial object centers, M: mixture of statistical shape models

Output: object segmentation $\mathbf{y}$

```
compute data cost D;
from object centers c_k set initial segmentation y and model shape parameters m_k;
compute shape cost V;
while not converged do
    update object pose parameters c_k and Θ_k ;
    if significant change of center c_k position, orientation Θ_k and object area then
        update remaining object shape parameters m_k;
        update shape costs V for all s close to k;
    end
    find superpixels ∂S_k on the external object boundary of k;
    optimize (7.10) wrt y by changing s ∈ ∂S_k using the greedy or Graph Cut algorithms;
end
```

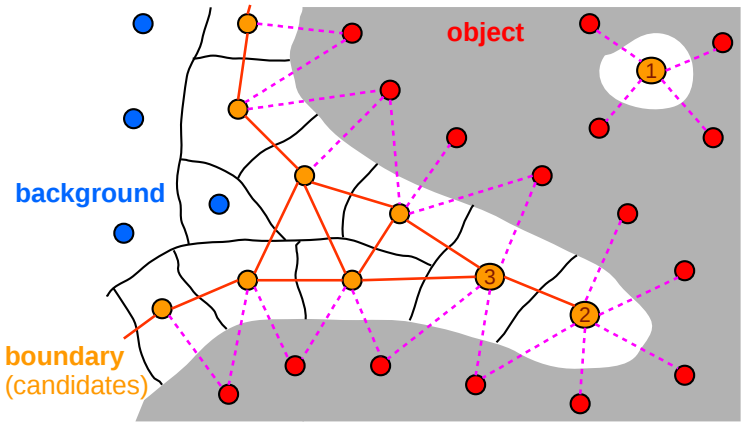
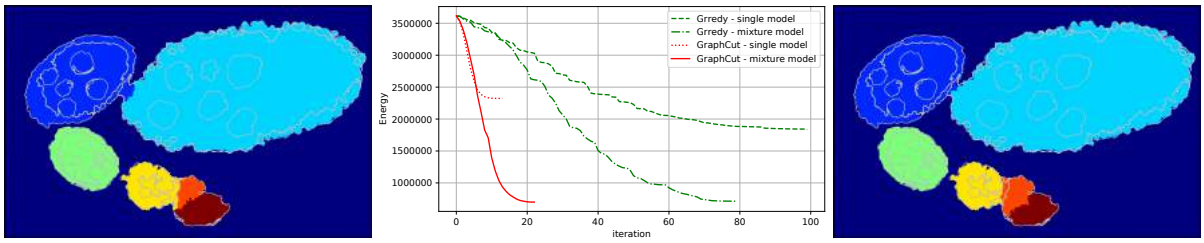


Figure 7.5: Creating a graph from ∂S_k on the boundary of object S_k. We connect all candidates of being objects neighboring superpixels ∂S_k (orange). For purposes of compactness, we also connect the neighboring object S_k (red) superpixels. This configuration imply pairwise penalty and impose the object compactness, see e.g. $s \in \{1,2\}$.

Greedy approach

We define a priority queue containing background superpixels s from $\partial S = \cup_k \partial S_k$ sorted by the energy improvement ΔE_s^k obtained by switching s to object k, which is a neighbor of s. A superpixel s is removed from the top of the queue if the energy improvement ΔE_s^k is positive, it is switched to the object label k, the model m_k is updated, and also the energy improvement ΔE of all superpixels neighboring with object k. The convergence can be accelerated by processing several best superpixels from the top of priority queue at once. This threshold can be a fixed



(a) (b) (c)

Figure 7.6: Example of region growing segmentation results applying greedy approach *(a)* and Graph Cuts *(c)*. We also show the energy evolution during iterations *(b)*. Each color region corresponds to an object, the thin white contours are leves of the appearance probabilities P_g.

number of superpixels or relative energy improvement, switching s where $\Delta E_s^k > \varepsilon E$. Note, the condition $\Delta E_s^k > 0$ still holds and once assigned s to an object k in particular sub-procedure can not be assigned later again to another object l. Regarding optimality and growing strategy, this number of assigned s in single iteration should be small.

Multiclass Graph Cut approach

attempts to find optimal labels $\mathbf{y}_s$ for superpixels from ∂S, the remaining labels are fixed. We create a graph from the superpixels $\bar{S}$ with edges connecting neighbors. We set the potentials from (7.10). A superpixel may only get a value of one of its neighboring object or a background. Other changes are forbidden by setting the corresponding unary potential to ∞. For optimizing this graph problem the standard $\alpha\beta$-swap Graph Cut algorithm is used [98].

Binary Graph Cut.

As a simplification, binary Graph Cut considers that a background superpixel $s \in \partial S_k$ neighboring with a single object k can either remain background or be switched to the label of its neighbor (see Figure 7.5). The modified unary and binary energy terms are obtained by a restriction of the general formulation (7.10) to the two possibilities for each s. The advantage of this formulation is that finding a global minimum is guaranteed and can be done quickly. We perform this binary Graph Cut sequentially on all objects and so it is convenient to allow swapping object labels, description follows.

Swapping object labels.

If two objects k and l touch during the optimization process, we found it useful to allow the superpixels on the boundary to exchange labels, thus shifting the border to reflect the shape models, even after the two objects have touched (see iterations in Figure 2.2 and results in Figure 7.10). It is implemented by adding these boundary pixels to the set ∂S.

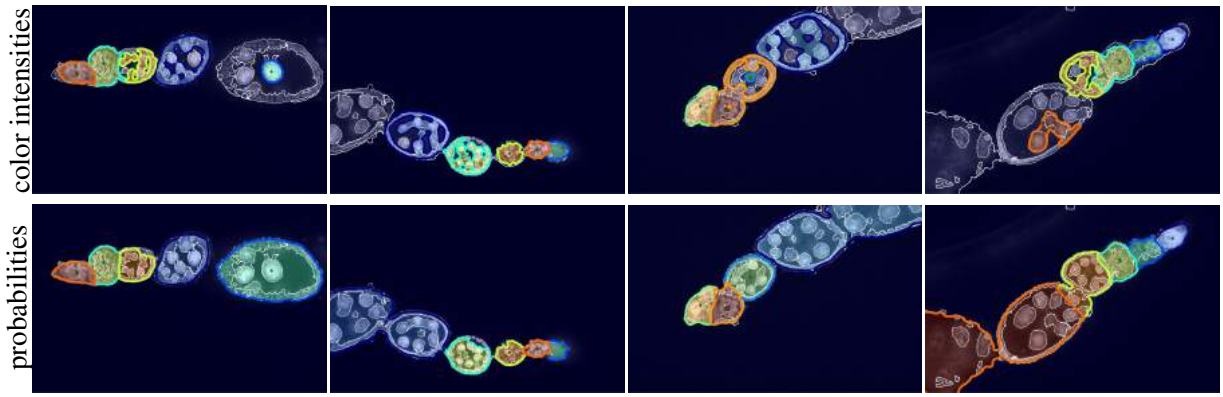


Figure 7.7: Examples of resulting segmentation using Morphological snakes on input images directly *(top row)* and on appearance probabilities P_g *(bottom row)*. The manual annotation for these images is presented in Figure 7.10.

7.3 Experiments

For all experiments we used the following SLIC parameters: superpixel size $\nu = 20$ pixels and regularization $\xi = 0.3$. The average running time to calculate the superpixels for our images of size 1000×1000 pixels was about 1 second.

The proposed method has a few parameters to set — the coefficient β and γ, and the update thresholds in Algorithm 1. Experimentally, we found that setting $\beta = 2$ and $\gamma = 5$ gives the best results for our images. We set the threshold for a shift to 20 pixels (superpixel size), rotation 10° degrees, and volume change to 5%. These values allow to reach the same segmentation quality as with updating V in every iteration, about twice as fast.

7.3.1 Region growing alternatives

We apply all methods on the results of the preliminary four class segmentation Y (see an example in Figure 2.1) as there is no method that can well segment individual eggs in our microscopy images of Drosophila ovary directly. For comparison we chose such methods to cover wide range of segmentation approaches that can be potentially used for this task — object segmentation, as we discussed in Section 2.3.

Watershed segmentation [57, 172, 173] is widely used for separating touching objects. We start from the binarized segmentation [38, 56] and apply the distance transform to calculate the distance of each pixel to the background. The watershed algorithm starting from initial centers is then used to identify individual objects. We also tested some morphological operations such as opening, before applying Watershed to see the improvement of the egg separation. It then turned out that selecting a universal structure element (SE) for all images is not reasonable because of (i) the large variance in egg size and (ii) connection thickness in between two eggs — a small SE does not always split neighboring eggs and a large SE may suppress the appearance of small

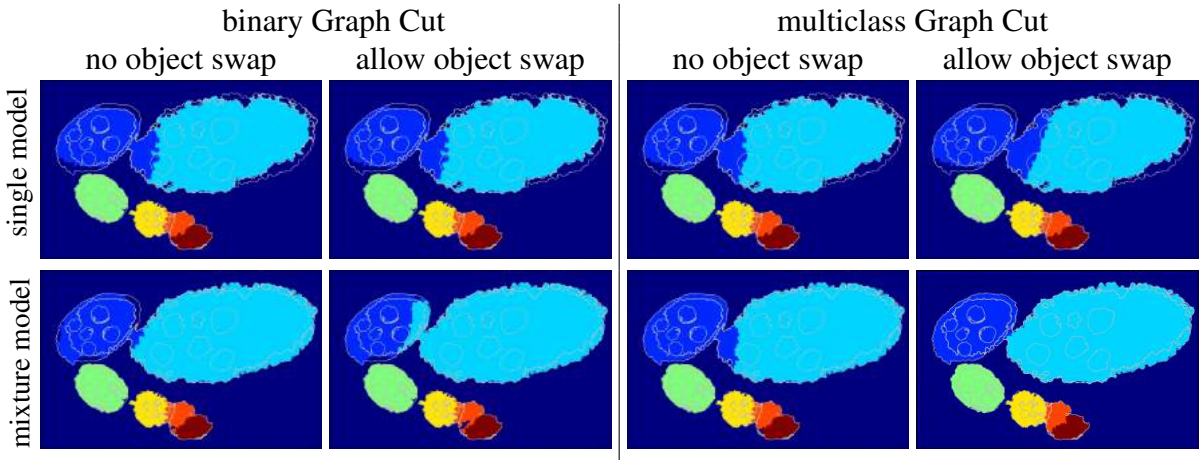


Figure 7.8: Resulting segmentation for several different variants of our method: single Gaussian model *(top row)* versus GMM *(bottom row)*, the binary and multiclass Graph Cut on the left and right half respectively. Colored regions represent individual objects and white levels the contours or segmentation Y.

eggs. We remark that in the experiments, we used morphological opening with the circular SE with 25 pixels in diameter.

Morphological Snakes [174] We used multiple morphological snakes with smoothing 3 and $\lambda_{1,2} = 1$ initialized from circular region around center with diameter 20 pixels which are approximately size of used superpixels evolving in parallel. We also adopted a restriction, that individual snakes cannot overlap. We apply the multi-snakes approach on the input image directly and also on appearance probabilities P_g, see Figure 7.7. The snakes on raw image frequently struggle with handling internal egg structure, on the other hand, snakes on P_g have difficulty to separate touching eggs.

Pixel-level Graph Cuts [78] optimize an energy function similar to the previous method but at the pixel level. The data term is distributed from superpixels to pixels in a straightforward way, $D_i(k) = -\log P_g\big(k \mid g(s(i))\big)$ for all pixels $i \in \Omega$ and standard pairwise regularization for Potts model. Pixels from a small region around the provided initial object centers c_k are forced to class k.

Superpixel-level graphcut [100] works similarly as above except that we assign classes k to superpixels, not pixels. The energy function $E(g)$ from (7.8) can be used directly, again without the shape cost V and without employing region growing.

method	model	obj. swap	Jaccard	accuracy	F_1 score	precision	recall
greedy	single	no	0.6433	0.9324	0.9324	0.9324	0.9324
greedy	single	yes	0.6367	0.9299	0.9299	0.9299	0.9299
greedy	mixture	no	0.7377	0.9583	0.9583	0.9583	0.9583
greedy	mixture	yes	**0.7527**	0.9577	0.9577	0.9577	0.9577
GC	single	no	0.6426	0.9317	0.9317	0.9317	0.9317
GC	single	yes	0.6220	0.9284	0.9284	0.9284	0.9284
GC	mixture	no	0.7360	0.9573	0.9573	0.9573	0.9573
GC	mixture	yes	**0.7544**	0.9568	0.9568	0.9568	0.9568

Table 7.1: Quantitative evaluation of the segmentation quality for several configurations of our region growing method.

superpixel size [px]		10	15	20	25	30	35	40
greedy	time [s]	1468	225	98	72	38	32	27
	Jaccard	0.755	0.754	0.753	0.753	0.752	0.746	0.741
Graph Cut	time [s]	94	41	21	9	7	6	5
	Jaccard	0.756	0.755	0.754	0.754	0.753	0.748	0.743

Table 7.2: Dependency of running time on superpixels sizes (respectively number of superpixels) with regularization $\xi = 0.3$. Note, the code has not been yet optimized for speed.

7.3.2 Comparison of region growing variants

We compare the different variants of our segmentation method: using Graph Cut vs greedy approach, GMM (with $M = 15$) vs single Gaussian (assuming GMM with $M = 1$), allowing object label swapping. Quantitative results are shown in Table 7.1. It confirms our observation (see Figure 7.8) that it is best to use a GMM with multiclass Graph Cut and label swapping with respect to Jaccard index which well reflects our visual observation.

Let us discuss the behavior of the Graph Cut and greedy region growing algorithms. The resulting segmentation of both Graph Cut and greedy are very similar. Speaking about the second criterion - processing time, the Graph Cut is faster in terms of a number of the iterations (see Figure 7.6(b)), but each iteration is little longer. The total processing time of Graph Cut approach is about 9 seconds compare to Greedy which takes about 72 seconds per image. We experimented with superpixel sizes and observed that they do not have a large influence on the segmentation quality, but they have a significant impact on the processing time, see Table 7.2. Region growing speeds up with larger superpixels and consequently fewer candidates to evaluate.

We also experimented the dependence of the resulting segmentation on the position of the initial centers c_k. We found that our method is very robust to the initialization — for a center initialization up to $1/2$ distance between the true center and the object boundary, we obtained visually equivalent results, see Figure 7.9.

	Jaccard	accuracy	F_1 score	precision	recall	time [s]
Watershed	0.5705	0.9246	0.9246	0.9246	0.9246	**5**
Watershed (w. morph.)	0.5705	0.9270	0.9198	0.9136	0.9327	7
Morph. snakes (image)	0.4251	0.8769	0.8070	0.9053	0.7987	784
Morph. snakes (P_g)	0.6494	0.8812	0.8812	0.8812	0.8812	968
Graph Cut (pixel-level)	0.7143	0.9204	0.9204	0.9204	0.9204	15
Graph Cut (superpixels)	0.3164	0.8643	0.8643	0.8643	0.8643	**3**
RG2Sp (greedy)	**0.7527**	**0.9577**	**0.9577**	**0.9577**	**0.9577**	72
RG2Sp (Graph Cut)	**0.7544**	**0.9568**	**0.9568**	**0.9568**	**0.9568**	9

Table 7.3: Quantitative comparison of the proposed region growing method (RG2Sp) with other baseline methods. We color the best (blue) and the second best (cyan) result.

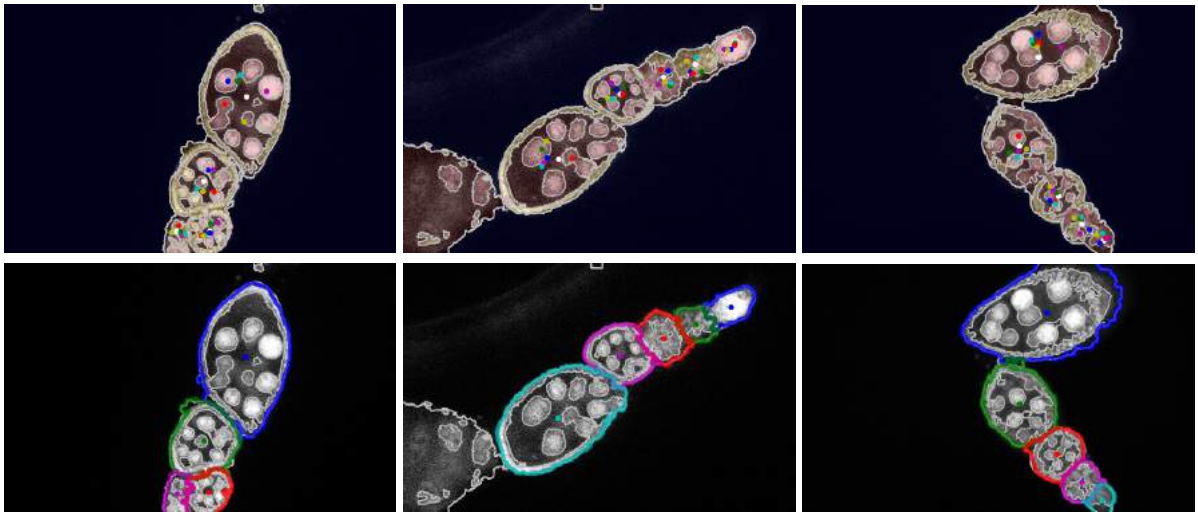


Figure 7.9: The impact of quality of initial center selection *(top row)* on the final segmentation *(bottom row)*. Each set of initial centers (colored equally for all eggs) was obtained by adding random displacement regarding particular egg size. For all initialization the region growing converged to the same segmentation.

7.3.3 Comparison with baseline methods

In the final experiment, we compare the performance of our selected method, i.e., region growing with a GMM, multiclass Graph Cut, and label swapping, and compare it with alternative baseline methods. Table 7.3 presents the quantitative results. We can see that proposed method performed better than the other methods in all comparable metrics.

Example segmentation results are shown in Figure 7.10. We can see that the alternative methods usually fail to properly distinguish touching eggs. Also, they are frequently merging two eggs together even if the second egg does not contain an initial seed which can happen in real-world application. It also may happen when the touching egg is not complete, so the object center detection method presented in Chapter 6 does not place a center inside such egg.

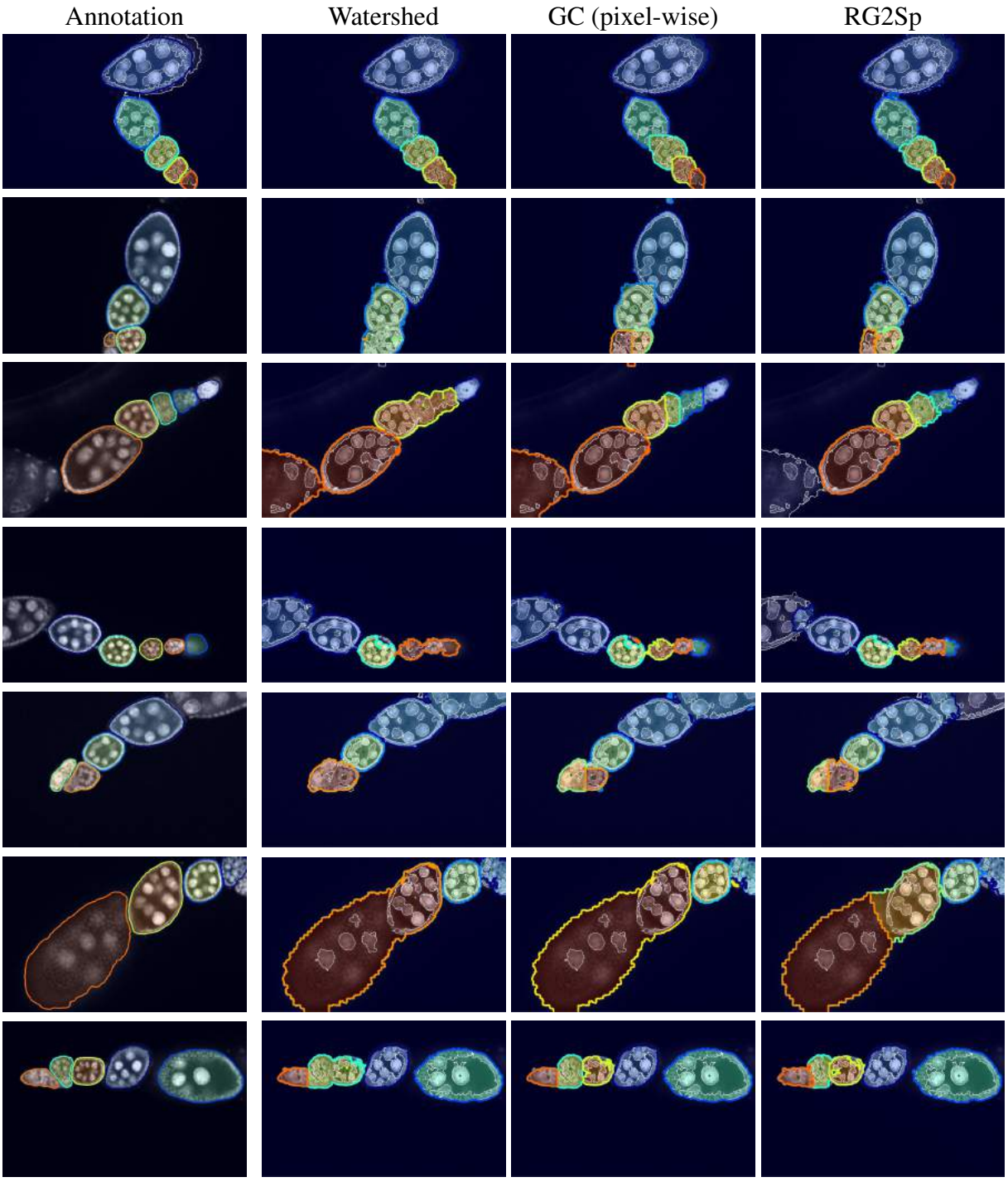


Figure 7.10: Each row represents a microscopy image segmented by an expert ('Annotation') and the three automatic methods — from left to right: watershed, Graph Cut on pixels and region growing. The expert annotation is shown overlaid on the input image. The segmentation result are shown overlaid over the input image with the preliminary four-class segmentation contours shown as thin while lines.

7.4 Summary

We presented a new region growing segmentation technique. It is fast thanks, tousling superpixels, and it is also robust thanks to handling the growing with Graph Cut and a ray feature based shape model. It can handle touching objects as well as objects with only partly visible boundaries. Compare to standard region growing method or other alternatives methods show better performances with comparable processing time. Another advantage of this method is a simpler generalization for other application, whenever a set of objects with known shapes is to be segmented.

8

Binary pattern dictionary learning

This chapter describes the final step of the complete pipeline (as illustrated in Figure 8.1 and Figure 1.3) for processing Drosophila gene expression images. This step consists of estimating a dictionary of atomic gene activations, so that each activation pattern can be well approximated by a union of a small sets of the so-called 'atomic-patterns' from this dictionary. We call this method Binary Pattern Dictionary Learning (BPDL). We illustrate the particular step in microscopy images of Drosophila imaginal discs as they are more intuitive.

We assume that the events we aim at are binary regions and can be expressed as a union of a small set of non-overlapping spatial patterns — so-called 'atlas'. Using such atlas and sparse binary encoding of input images reduces the size of stored information and automatically poses some spatial relations. This leads itself well to further automatic analysis, with the hope of discovering new biological relationships. Later in Section 8.4, we compare proposed BPDL methods to existing alternative linear decomposition methods on synthetic data and we also show results of these algorithms on real microscopy images of the Drosophila imaginal discs.

This method expects only single object in the image which is natural in images of an imaginal disc (or can be easily extracted, e.g., thresholding), but for images or ovary, the objects have to extracted approximately by an ellipse (see Chapter 6) or fine by region growing presented in Chaper 7. For extracting the gene activation we use segmentation from Chapter 5.

This chapter is strongly based on the paper [44]. Several images were enlarged, and some notations were changed to unify with the rest of the thesis. The part describing superpixel segmentation was reduced since it has been presented in Chapter 5.

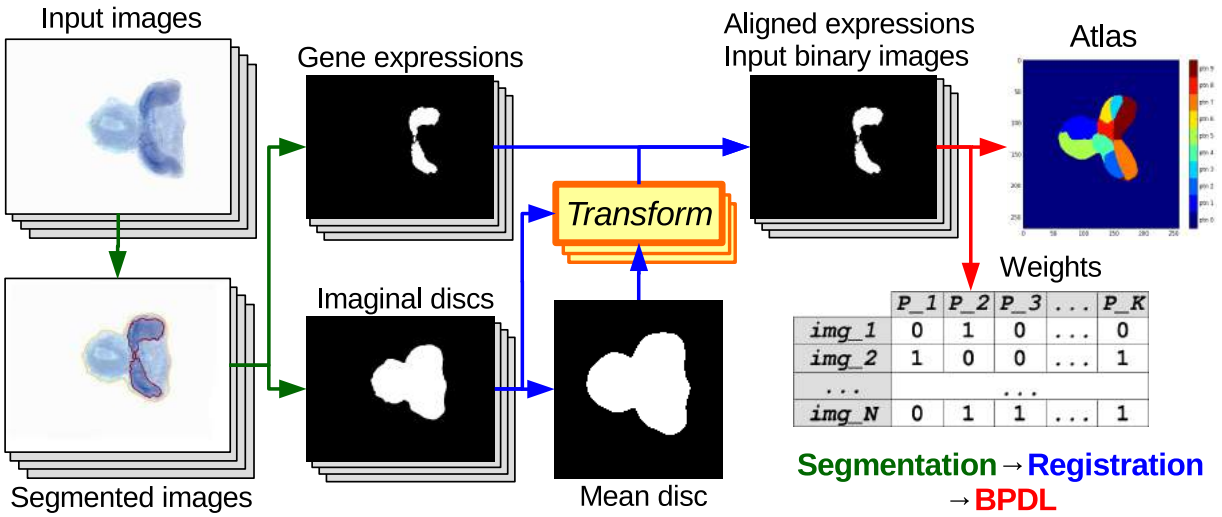


Figure 8.1: A flowchart of the complete pipeline for processing images of Drosophila imaginal discs: (i) unsupervised superpixel segmentation into 3 classes (background, imaginal disk, and gene expression); (ii) registration of binary segmented imaginal discs onto a reference shape (disc prototype); (iii) atlas estimation from aligned binary gene expressions.

8.1 Preprocessing

Let us briefly review the processing applied on input images which was introduced in previous chapters and it required to obtain a suitable set of input images.

Given a set of images of imaginal discs (see Figure 3.2a) containing both anatomical and gene expression information, preprocessing is applied to obtain a set of segmented and aligned gene expression images, which serves as input for the subsequent binary dictionary learning (see Figure 8.1 or more general Figure 1.3).

Segmentation. We use unsupervised superpixel segmentation proposed in Section 5: First, SLIC superpixels are calculated [87] with an initial size of 15 pixels. For each superpixel, color features are computed. Then, the superpixels are assigned to one of the following four classes (background, imaginal disc or gene activations) using GMM with EM algorithm assuming that each Gaussian represents the single class with Graph Cut regularization. As we run unsupervised segmentation, we perform a post-processing — identifying the imaginal disc component and fill holes in the union of imaginal disc with gene activations as imaginal disc.

Registration. For each of the four disc types, a reference shape is calculated as the mean disc shape over all images. Then all other images are registered to the reference shape. We use a fast elastic registration [43], which works directly on the segmented images, transforming the disc shapes by aligning their contours, ignoring the activations (see Figure 3.2*top*). The activations (Figure 3.2*bottom*) are then aligned using the recovered transformation.

8.2 Problem definition

Let us update the general formulation of a linear decomposition, presented in Section 2.4, of input images $\mathbf{G}$ and formulate as a binary problem [151] which can be found by minimizing

$$\min_{Y',W} \left\| \mathbf{G} - Y' \cdot W \right\|^2 \tag{8.1}$$

where set of images is represented as a matrix $\mathbf{G} \in \{0,1\}^{|\Omega| \times N}$ — a rearranged set of image pixels of N images with pixel coordinates Ω, $Y' \in \{0,1\}^{|\Omega| \times \mathbb{L}}$ corresponds to an atlas with L patterns and $\mathbf{W} \in \{0,1\}^{L \times N}$ are image specific weights.

Pattern dictionary learning is an approximation of sequence or images samples over time or instances by small number of spatial patterns such that each input image can be represented (reconstructed) by a union of a few patterns from this dictionary — so-called 'atlas'.

Our task is estimation of an atlas (segmentation) of unique compact patters (region sharing the same labels) from a set of aligned binary images — binary segmentation of gene activation segmented in Chapter 5, where each input image is coded by a binary vector marking present of a particular pattern in the image. Note that the alignment is according to object boundary, not segmented gene activation. The main difference to the linear decomposition that our task is constrained to be binary on outputs.

Let us define the image pixels as $\Omega \subseteq \mathbb{Z}^d$, with $d = 2$, and the input binary image as $g : \Omega \to \{0,1\}$. Our task is to find an atlas $\mathbf{y}_\Omega : \Omega \to \mathbb{L}$, with labels $\mathbb{L} = [0,\ldots,K]$, assigning to each pixel either a background (label $l = 0$), or one of the labels (patterns, set of equal labels) $1,\ldots,K$. Note that the atlas formulated as $\mathbf{y}_\Omega$ prevent overlapping patterns compare to Y' by default and the transfer from Y' to $\mathbf{y}_\Omega$ is straightforward. Each binary weight vector $\mathbf{w} : \mathbb{L} \to \{0,1\}$ yields an image $\hat{g}$ as a union of the selected patterns in atlas $\mathbf{y}_\Omega$

$$\hat{g}_i = \sum_{l \in \mathbb{L}} w_l \cdot [\![\mathbf{y}_i = l]\!] \tag{8.2}$$

where $[\![\cdot]\!]$ denotes the Iverson bracket. Note, using this representation where the patterns cannot overlap the union can be replaced by a simple sum The approximation error on one image g and its representation by $\mathbf{y}_\Omega$ and $\mathbf{w}$ is the Hamming distance

$$F(g, \mathbf{y}_\Omega, \mathbf{w}) = \sum_{i \in \Omega} [\![g_i \neq \hat{g}_i]\!] = \sum_{i \in \Omega} \left| g_i - \sum_{l \in \mathbb{L}} w_l \cdot [\![\mathbf{y}_i = l]\!] \right| \tag{8.3}$$

where w_l is a binary value.

To encourage spatial compactness of the estimated atlas, we shall penalize class differences between neighboring pixels i, j in the atlas

$$H(\mathbf{y}_\Omega) = \sum_{\substack{i,j \in \Omega, i \neq j, \\ d(i,j)=1}} [\![\mathbf{y}_i \neq \mathbf{y}_j]\!] \tag{8.4}$$

Algorithm 2: General schema of BPDL algorithm.

```
initialize atlas y_Ω;
while not converged do
    update weights W;
    reinitialize empty patterns in y*_Ω;
    update atlas y*_Ω via Graph Cuts;
end
```

where the Kronecker delta $[\![\mathbf{y}_i \neq \mathbf{y}_j]\!]$ for all combinations of $\mathbf{y}_{i,j} \in \mathbb{L}$ can be represented as a square matrix with zeros on the main diagonal and ones otherwise. In other words, we pay zero penalties if neighboring pixels belong to the same class and a positive value otherwise, according to assigned labels $\mathbf{y}_i$ and $\mathbf{y}_j$, and given matrix.

The optimal atlas and the associated weights are found by optimizing the mean approximation error for all N images plus a spatial regularization

$$\mathbf{y}^*_\Omega, \mathbf{w}^* = \arg\min_{\mathbf{y}_\Omega, \mathbf{W}} \frac{1}{N} \sum_n F(g^n, \mathbf{y}_\Omega, \mathbf{w}^n) \; + \beta \cdot H(\mathbf{y}_\Omega) \tag{8.5}$$

where the matrix $\mathbf{W}$ contains all weights $\mathbf{w}^n$ for $n \in [0, \ldots, N]$, and β is the spatial regularization coefficient. Sufficiently large β encourages the patterns to be smooth and connected.

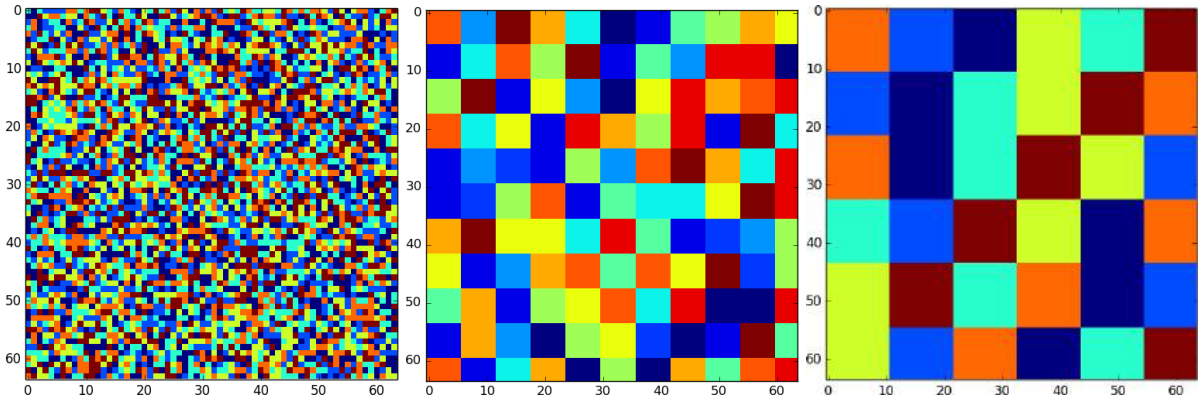
8.3 Alternating minimization

The criterion (8.5) is minimized alternately with respect to atlas $\mathbf{y}_\Omega$ and weights $\mathbf{W}$, (see Algorithm 2). For lack of week initialization and potential collapsing some patterns, we add reinitialization step to this loop.

Initialization. We initialize the atlas with randomly labeled patches on a regular grid, with user-defined sizes; see Figure 8.2 for examples.

Update weights W. With the atlas $\mathbf{y}_\Omega$ fixed, we estimate the weights $\mathbf{w}^n$ for each image g^n independently. It turns out that $F(g^n, \mathbf{y}_\Omega, \mathbf{w}^n)$ is minimized with respect to w^n_l, if the majority of pixels in the pattern $l \in \mathbf{y}_\Omega$ agree with positive activation the image. It is clear that for centain image g and pattern l the criterion is minimal $F(g^n, \mathbf{y}_\Omega, 1) < F(g^n, \mathbf{y}_\Omega, 0)$ if the majority of pixels inside the pattern l are also positive valued in the sense image g $\sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i = 1]\!] > \sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i \neq 1]\!]$ and vice-versa. Accordingly, we set

$$w_l = [\![Q(g, \mathbf{y}_\Omega, l) \geq \sigma]\!] \tag{8.6}$$



(a) (b) (c)

Figure 8.2: Random atlas initialization with patch sizes 1 pixel *(a)*, $m/2K$ pixels *(b)*, and m/K pixels *(c)*, for $K = 6$ and $m = 64$ pixels being the image size.

where $\sigma = 1$ and

$$Q(g, \mathbf{y}_\Omega, l) = \frac{\sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i = 1]\!]}{\sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i \neq 1]\!]} \tag{8.7}$$

$$= \frac{\sum_{i \in \Omega} [\![\mathbf{y}_i = l]\!] - \sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i \neq 1]\!]}{\sum_{i \in \Omega, \mathbf{y}_i = l} [\![g_i \neq 1]\!]} = \frac{\sum_{i \in \Omega} [\![\mathbf{y}_i = l]\!]}{\sum_{i \in \Omega, \mathbf{y}_i = l} (1 - g_i)} - 1 \tag{8.8}$$

We temporarily reduce σ in the initial stage of the Algo. 2, otherwise very few patterns might be selected.

Reinitialize empty patterns. After the binary weight calculation step, some patterns (homogeneously labeled regions in $\mathbf{y}_\Omega$) may not have been used for any image. This is wasteful, unless the reconstruction is already perfect, we can always improve it by adding another pattern. We iterate the following procedure until all K labels are used:

1. find an image g^n with the largest unexplained residual $(g^n - \hat{g}^n) > 0$
2. find the largest connected component of this residual and assign label $l \notin \mathbf{y}_\Omega$;
3. calculate weights w_l^n for the new label l for all images $g^n \in \mathbf{G}$ using (8.6).

Update of atlas $\mathbf{y}_\Omega$. With the weight vectors $\mathbf{W}$ fixed, finding the atlas $\mathbf{y}_\Omega$ is a discrete labeling problem. We can rewrite the criterion in (8.5) as

$$\sum_{i \in \Omega} \underbrace{\frac{1}{N} \sum_n \left| g_i^n - \sum_{l \in \mathbb{L}} w_l^n \cdot [\![\mathbf{y}_\Omega = l]\!] \right|}_{U(y_i)} + \beta \sum_{\substack{i,j \in \Omega, i \neq j, \\ d(i,j)=1}} \underbrace{[\![\mathbf{y}_i \neq \mathbf{y}_j]\!]}_{B(\mathbf{y}_i, \mathbf{y}_j)} \tag{8.9}$$

which can be solved for example with Graph Cut [98] and alpha expansion.

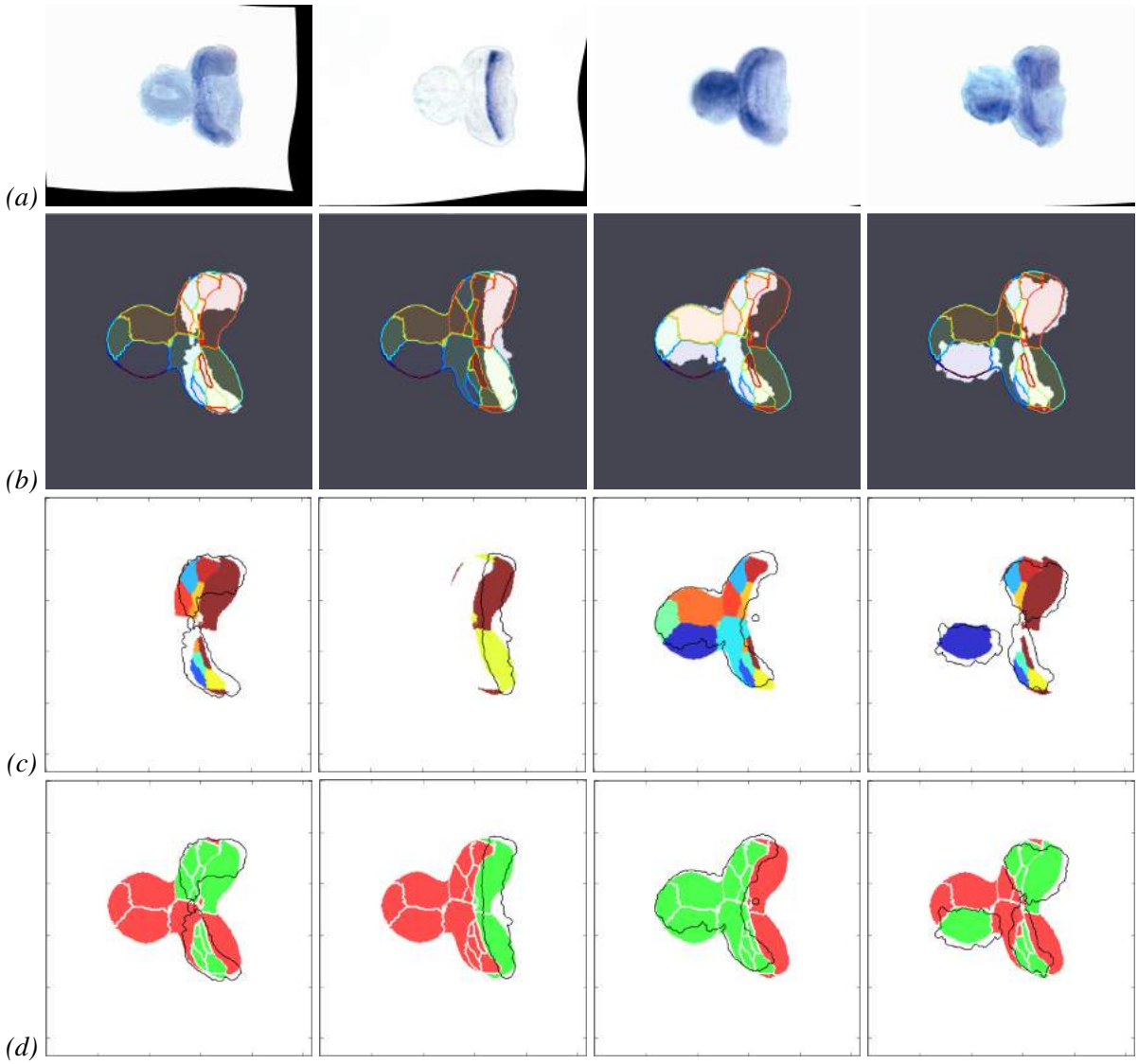


Figure 8.3: Visualization of the reconstruction of real images (already shown in Figure 3.2) in three different ways: *(a)* The original registered image. *(b)* The input binary segmentation of gene expression overlapped by the atlas pattern contours. *(c)* The individual atlas patterns (in color) with the binary input gene expression segmentation overlaid (black contour). *(d)* Used (green) versus unused (red) atlas patterns with contour of the segmented input expression boundary (black).

8.4 Experiments

We evaluate the performance (atlas similarity and descriptiveness and elapsed time) of the algorithm on both synthetic and real images.

8.4.1 Dictionary learning alternatives

We have compared our BPDL with the following methods: NMF [149], FastICA [147], SparsePCA [146] and Dictionary Learning [148] (DL). All methods are implemented in the scikit-learn [166] library.

Binarization of continues components. We want to compare two approaches, linear and binary components While the task is estimating binary patterns, we need to binarize the linear components to be all in the same form. In previous work [156, 175] the binarization of linear components was performed by thresholding. To obtain a binary atlas $\mathbf{y}$ from a continuous matrix $Y \in \mathbb{R}^{|\Omega| \times L}$, we select the component with a maximal value

$$\mathbf{y}_i = \arg\max_{l \in \mathbb{L}} Y_i^l$$

in each pixel position $i \in \Omega$.

8.4.2 Comparison on synthetic datasets

In Table (8.1) we show the accuracy of reconstructing the atlas (measured by ARS), the mean approximation error R, and elapsed time for all datasets and their modifications. The number of patterns was set to the true value K.

We can say that on the 'pure' images, all methods work well. In other cases, the accuracy of our method (as measured by ARS and R) is better. The fastest method is the NMF (on average twice as fast as BPDL) but its results are poor. On the other hand, FastICA gives the second best quality results after BPDL but is much slower (on average 40 times slower then BPDL).

8.4.3 Comparison on real images

We applied all methods on segmented gene expressions images of the Drosophila imaginal discs varying the number of patterns $K \in \{10, 20, 30\}$. Several reconstruction examples for BPDL are shown in Figure 8.3. Looking at the estimated atlases (Figure 8.4) we found that NMF, FastICA and DL have difficulty to identify background and often produce very small regions. Example atlases by BFDL on all four considered disc types are shown in Figure 8.6.

The effect of the Graph Cut regularization parameter β is shown in Figure 8.5. A value of $\beta = 0.001$ was found to perform best by subjective evaluation and it was used in all other experiments.

8.5 Summary

This chapter addresses automatic image analysis of Drosophila imaginal discs, focusing on the problem of finding an atlas of atomic gene expression from the images. Unlike alternative methods, we assume that the atlas and its coefficients are binary and our proposed method

datasets		NMF	FastICA	sPCA	DL	BPDL
v0		*(size* 64×64 *px, 6 patterns)*				
pure	ARS	**1.0**	**1.0**	**0.961**	**1.0**	**0.999**
	error R	**0.0**	**0.0**	**0.002**	**0.0**	**0.0**
	time	**2.780**	168.476	30.842	304.51	**6.658**
deform	ARS	0.775	**0.921**	0.769	0.777	**0.993**
	error R	0.014	**0.004**	0.0213	0.014	**0.0**
	time [s]	**1.697**	141.527	22.833	279.87	**4.766**
D&N	ARS	0.048	**0.778**	0.002	0.066	**0.999**
	error R	0.033	**0.014**	0.033	0.033	**0.0**
	time [s]	**2.005**	229.47	24.907	598.83	**6.774**
v1		*(size* 64×64 *px, 13 patterns)*				
pure	ARS	**1.0**	**1.0**	0.992	**0.995**	**0.999**
	error R	**0.0**	**0.0**	0.0298	**0.019**	**0.0**
	time	**2.333**	340.32	18.291	737.47	**6.029**
deform	ARS	0.785	**0.948**	0.780	0.779	**0.992**
	error R	0.017	**0.004**	0.029	0.033	**0.005**
	time [s]	**4.001**	312.18	15.000	700.03	**7.561**
D&N	ARS	0.091	**0.878**	0.009	0.0727	**0.951**
	error R	0.048	**0.010**	0.061	0.0499	**0.003**
	time [s]	**4.490**	439.04	11.420	697.599	**9.562**
v2		*(size* 128×128 *px, 23 patterns)*				
pure	ARS	**1.0**	**1.0**	**0.989**	**1.0**	**0.999**
	error R	**0.0**	**0.0**	0.037	**0.0**	**0.005**
	time [s]	**82.329**	5533.4	460.82	14786.	**88.260**
deform	ARS	0.818	**0.846**	0.801	0.807	**0.970**
	error R	0.019	**0.015**	0.056	0.046	**0.004**
	time [s]	**144.10**	5683.2	477.47	13619.	**165.22**
D&N	ARS	0.120	**0.612**	0.024	0.144	**0.877**
	error R	**0.036**	**0.036**	0.092	0.039	**0.013**
	time [s]	**77.399**	6912.9	485.44	13729.	**289.51**

Table 8.1: Performance comparison on the synthetic datasets. We show the atlas ARS (Adjusted Rand Score), approximation error R by (3.1), and processing time in seconds. We color the best (blue) and the second best (cyan) result. All experiments were performed on the same computer, in a single thread configuration. The results show that all methods work well on the 'pure' subset. For the deformed and also noise images the best results were obtained by BPDL. The fastest method was NMF, followed by BPDL. Datasets v0, v1 and v2 are presented in Figure 3.4 and reflect difficulty levels.

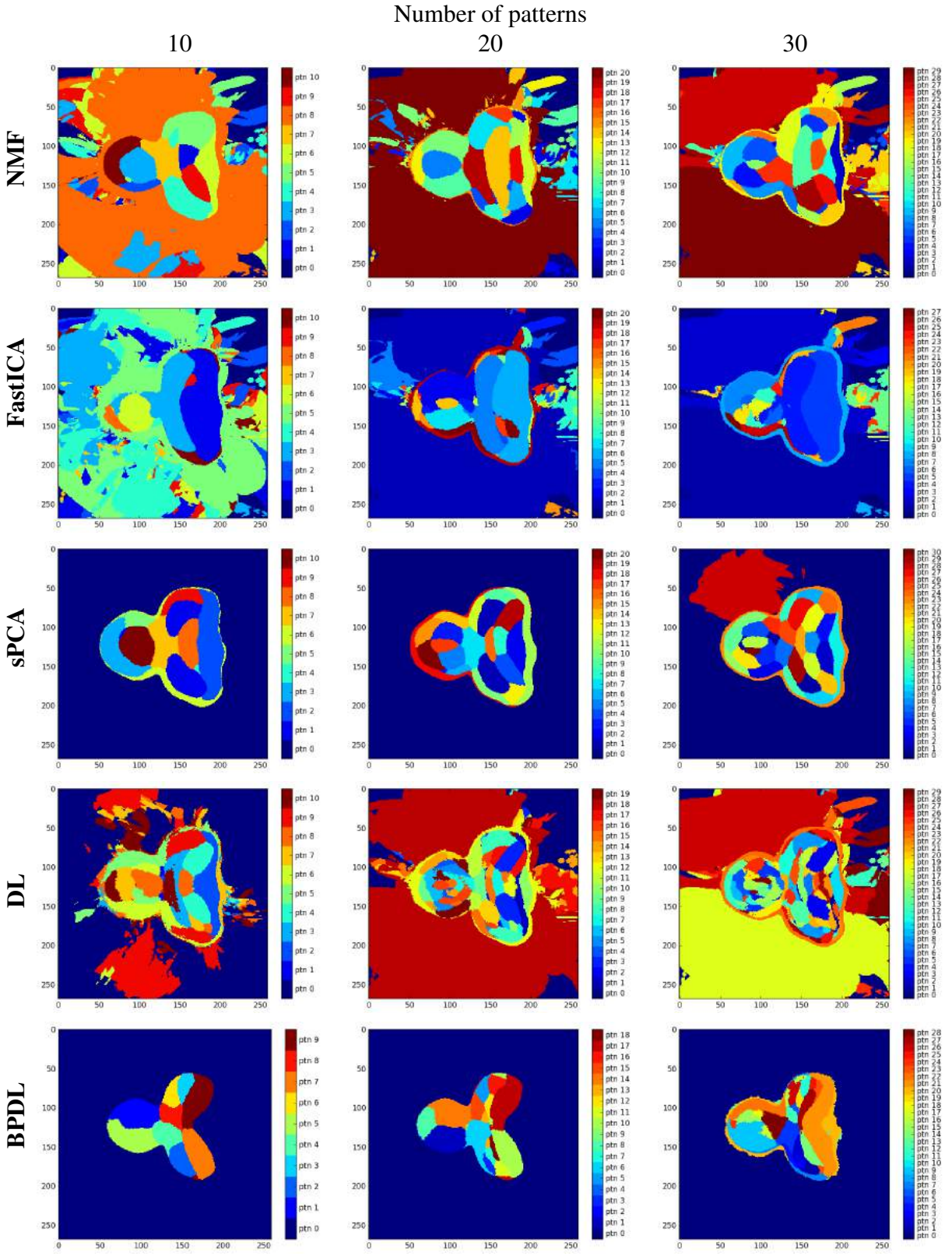


Figure 8.4: Presenting estimated atlases by all methods with different number of estimated patterns K.

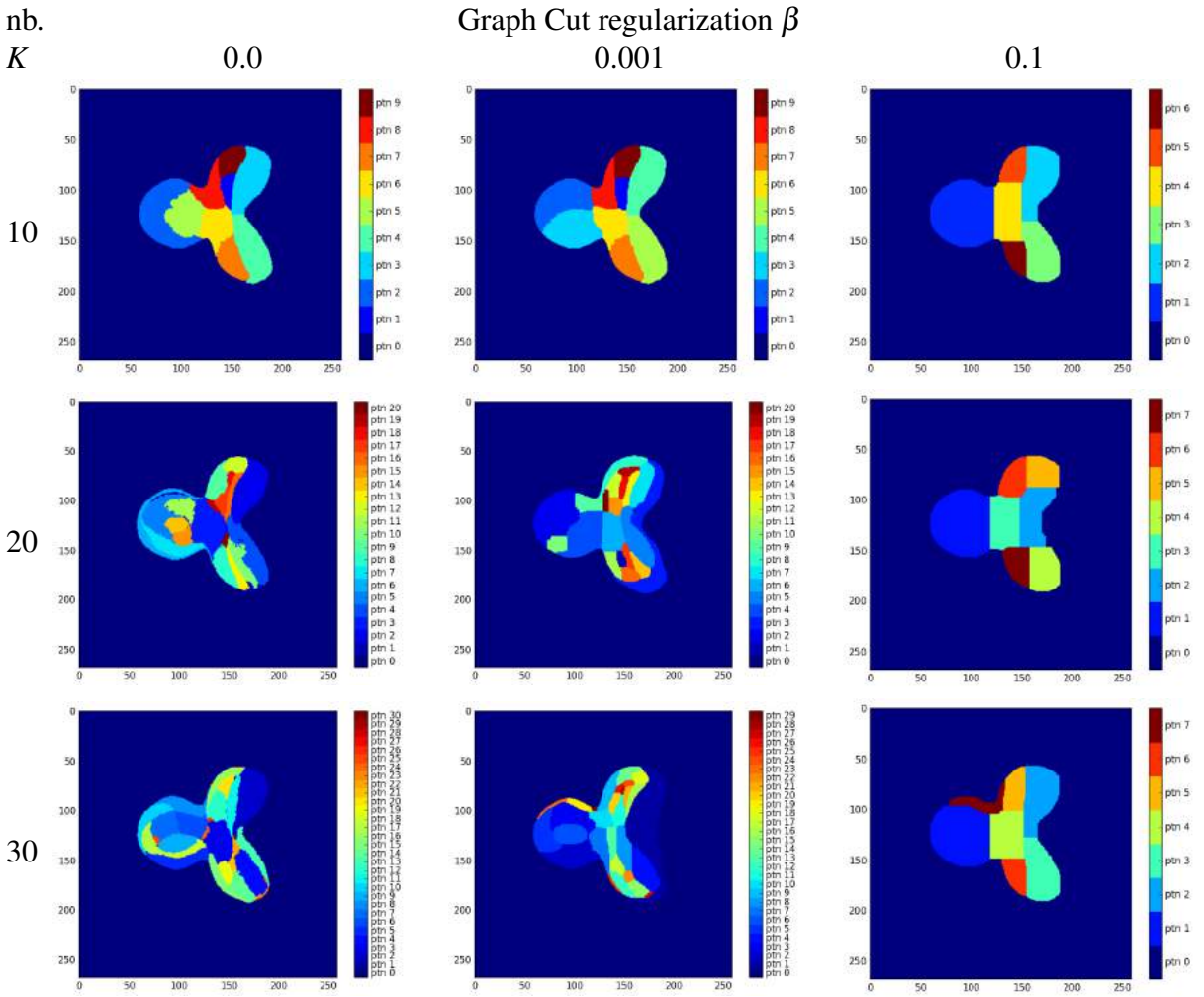
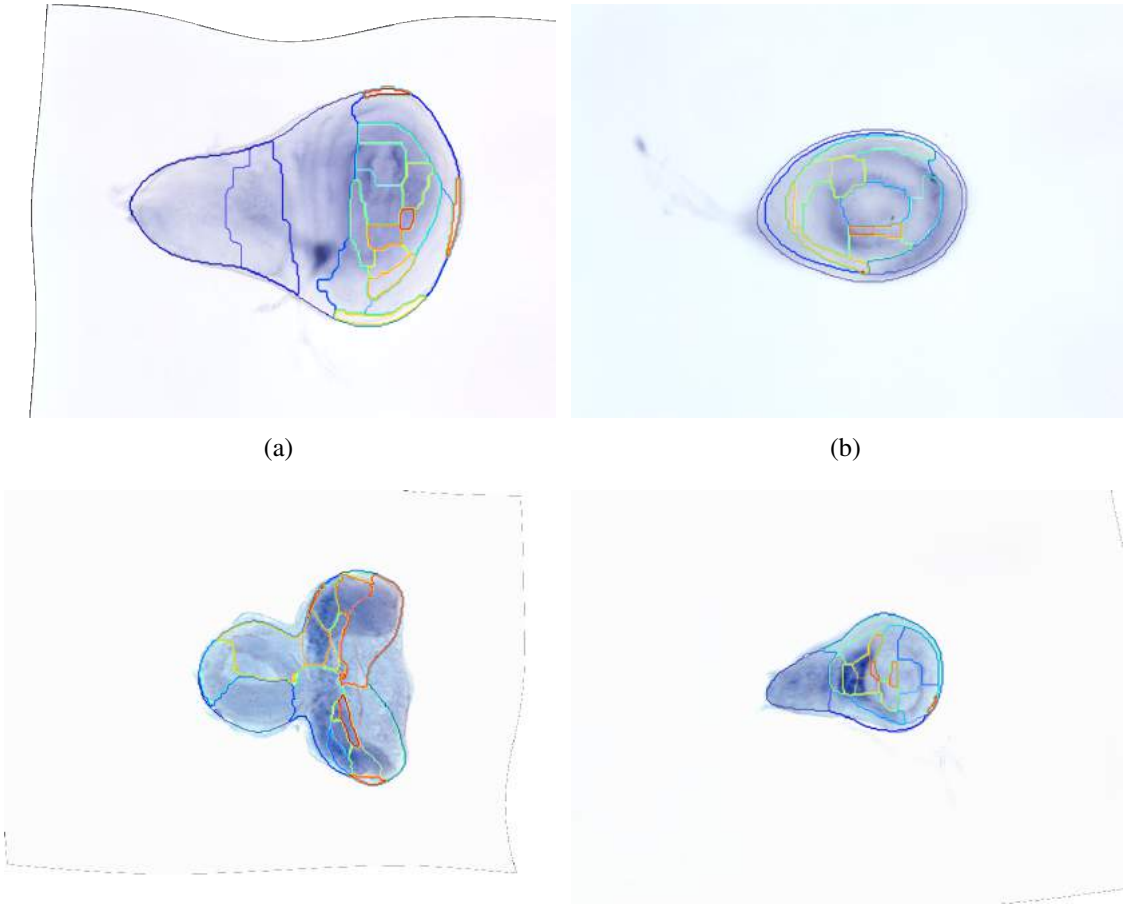


Figure 8.5: Visualization of the estimated atlas for Drosophila images (eye type imaginal discs) as a function of the number of estimated patterns K and the Graph Cut regularization parameter β.

(BPDL) estimates an atlas of binary patterns directly by an iterative procedure. On synthetic datasets, BPDL achieves the best overall quality results, with a very reasonable computational complexity. On real datasets, BPDL produces similar quality atlas and reconstruction as the SparsePCA method, while being much faster. The extracted image labels will be further processed by data mining methods. The proposed binary pattern dictionary learning can be applied any time a large set of binary images should be represented by a small dictionary.

Method	Number of patterns K			Time [min]
	10	20	30	
NMF	0.0939	0.0823	0.0723	**10**
FastICA	0.1197	0.0779	0.0485	24
sPCA	**0.0476**	**0.0413**	**0.0352**	477
DL	0.0939	0.0648	0.0596	338
BPDL	**0.0467**	**0.0395**	**0.0361**	**20**

Table 8.2: Reconstruction difference R on real images of imaginal disc (eye type) by all tested methods for three different assumed numbers of patterns K.



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

Figure 8.6: Sample images of each imaginal disc types: wing *(a)*, leg *(b)*, eye *(c)*, haltere *(d)* with the atlases estimated by BPDL shown as contour overlays for number of patterns $K = 20$.

9 Conclusions

With the growing amounts of available images together with their increasing resolution, a manual analysis by an expert becomes almost infeasible. The semantic segmentation of ovary is difficult to die to high similarity between two pairs of classes; the estimation of egg center have not been proposed yet; performing an instance segmentation for such highly structured object is however very challenging, and standard techniques for a pattern extractions tend to be sensitive to noise and internal deformations. In Chapter 3, we introduced used datasets with a variety of user annotation depending on the image processing task. Later, we proposed four image analysis methods and presented performances on real microscopy images and we compared them with standard methods.

The central objective of this work was to improve a fully automatic processing and analysis pipeline for microscope images of the Drosophila. In particular, the proposed methods are — image segmentation, object center detection, region growing for sensed individual images and followed by binary pattern extraction as atlas estimation from a broad set of sensed images. Our proposed enhancement to the superpixel extraction regarding processing time and smarter post-processing was published as a plugin to ImageJ. In chapter 4, we presented enhancement to superpixel extraction led to improved performance compared to standard pixel-wise methods on real microscopy images. Furthermore in Chapter 5, introducing Graph Cut regularization as a last processing step in the superpixel segmentation and by using our proposed model edge weight led to a further improvement in the segmentation. We showed that in some applications an unsupervised segmentation might have a very similar result as supervised segmentation. In Chapter 6, improvements of the object center detection resulted from our proposed novel features – label histogram and rotationally invariant ray features – as well as from clustering of center-point proposals. Approximating an egg through ellipse-fitting helped to significantly reduce multiple detections of a single egg object. Finally in Chapter 7, introducing region growing coupled with a shape prior learned from a small set of training examples provided increased ro-

bustness in handling touching objects as well as for dealing with inaccurate initial segmentation. For the last processing step presented in Chapter 8, the atlas estimation through binary pattern extraction, we assumed that such patterns should be compact and their coefficients binary. Our proposed method used an iterative procedure and estimated the atlas directly from the aligned input image set. In comparison to alternative decomposition methods, our method yielded higher resistance to internal deformations and input noise on both synthetic and real datasets.

These methods were developed as part of the large project in cooperation with MPI-CBG group from Germany. The complete pipeline was described in the beginning of this work - automatic image analysis pipeline for mining gene patterns from a vast set of microscopic images, see Figure 1.3. Possible continuation of this project is extending these methods to work in 3D entirely (note that spacing is $1 \times 1 \times 12$) and release it as close form software/plugin which can be simply used by a biologist.

Even though being developed for the particular application of Drosophila images, the proposed method can be easily generalized and adapted to other application domains, not only in the biomedical context and on microscopy images. To ease the access to the presented methods and also to encourage the community to possibly improve the proposed methods, the implementations, as well as the source code, were released as open-source for public usage.

Future Work

The author of the thesis suggests to explore the following:

- Extend whole image analyses pipeline on 3D volume as it is the original dimension of sensed microscopy images of Drosophila ovary.
- In BPDL, introduce deformation inside each image and investigate how to chose optimal number ob extracting patterns as a part of the optimization criterion.
- Experiment with Deep learning approaches for instance segmentation as it may solve the individual egg segmentation and stage classification jointly.

Bibliography

[1] A. M. Arias, "Drosophila melanogaster and the Development of Biology in the 20th Century," pp. 1–25, 2008.

[2] Medzhitov R., Preston-Hurlburt P., and Janeway C.A. Jr, "A human homologue of the Drosophila Toll protein signals activation of adaptive immunity," *Nature*, vol. 388, no. 6640, pp. 394–397, 1997.

[3] E. Bier, "Drosophila, the golden bug, emerges as a tool for human genetics," *Nature Reviews Genetics*, vol. 6, no. 1, pp. 9–23, 2005.

[4] L. L. Song, R. Liang, D. D. Li, and E. Al., "A Systematic Analysis of Human Disease-Associated Gene Sequences In Drosophila melanogaster," *Genome Research*, vol. 59, no. 23, pp. 1114–1125, 2001.

[5] C. J. Potter, G. S. Turenchalk, and T. Xu, "Drosophila in cancer research. An expanding role.," *Trends in genetics : TIG*, vol. 16, no. 1996, pp. 33–39, 2000.

[6] V. A. Rudrapatna, R. L. Cagan, and T. K. Das, "Drosophila cancer models," *Developmental Dynamics*, vol. 241, no. 1, pp. 107–118, 2012.

[7] L. He, X. Wang, and D. J. Montell, "Shining light on Drosophila oogenesis: live imaging of egg development," *Current Opinion in Genetics & Development*, vol. 21, no. 5, pp. 612–619, 2011.

[8] A. Jory, C. Estella, M. W. Giorgianni, M. Slattery, T. R. Laverty, G. M. Rubin, and R. S. Mann, "A Survey of 6,300 Genomic Fragments for cis-Regulatory Activity in the Imaginal Discs of Drosophila melanogaster," *Cell Reports*, vol. 2, no. 4, pp. 1014–1024, 2012.

[9] R. Bastock and D. St Johnston, "Drosophila oogenesis," *Current Biology*, vol. 18, no. 23, pp. 1082–1087, 2008.

[10] D. Kirilly and T. Xie, "The Drosophila ovary: an active stem cell community," *Cell Research*, vol. 17, no. 1, pp. 15–25, 2007.

[11] S. Roth and J. A. Lynch, "Symmetry Breaking During Drosophila Oogenesis," *Cold Spring Harbor Perspectives in Biology*, vol. 1, no. 2, pp. a001891–a001891, 2009.

[12] P. P. Marie, S. Ronsseray, and A. Boivin, "From Embryo to Adult: piRNA-Mediated Silencing Throughout Germline Development in Drosophila.," *G3: Genes | Genomes | Genetics*, vol. 7, no. February, pp. 1–37, 2016.

[13] J. V. Beira and R. Paro, "The legacy of Drosophila imaginal discs," *Chromosoma*, vol. 125, no. 4, pp. 573–592, 2016.

[14] J. N. Wilcox, "Fundamental principles of in situ hybridization.," *The journal of histochemistry and cytochemistry : official journal of the Histochemistry Society*, vol. 41, no. 12, pp. 1725–1733, 1993.

[15] C. Harmon, P. Ahammad, A. Hammonds, R. Weiszmann, S. Celniker, S. Sastry, and G. Rubin, "Comparative analysis of spatial patterns of gene expression in Drosophila melanogaster imaginal discs," *Proceedings of the International Conference on Research in Computational Molecular Biology*, pp. 1–15, 2007.

[16] M. Baker, "Cellular imaging: Taking a long, hard look," *Nature*, vol. 466, no. 7310, pp. 1137–1140, 2010.

[17] P. Tomancak, A. Beaton, R. Weiszmann, E. Kwan, S. Shu, S. E. Lewis, S. Richards, M. Ashburner, V. Hartenstein, S. E. Celniker, and G. M. Rubin, "Systematic determination of patterns of gene expression during Drosophila embryogenesis.," *Genome biology*, vol. 3, no. 12, p. RESEARCH0088, 2002.

[18] G. M. Rubin and a. C. Spradling, "Genetic transformation of Drosophila with transposable element vectors.," *Science (New York, N.Y.)*, vol. 218, no. 4570, pp. 348–53, 1982.

[19] W. Driever and C. Nusslein-Volhard, "The bicoid protein determines position in the Drosophila embryo in a concentration-dependent manner," *Cell*, vol. 54, no. 1, pp. 95–104, 1988.

[20] B. A. Edgar and P. H. O'Farrell, "Genetic Control of Cell Division Patterns in the Drosophila Embryo," *Cell*, vol. 57, no. 1, pp. 177–187, 1989.

[21] K. McCall, "Eggs over easy: Cell death in the Drosophila ovary," 2004.

[22] P. Tomancak, B. P. Berman, A. Beaton, R. Weiszmann, E. Kwan, V. Hartenstein, S. E. Celniker, and G. M. Rubin, "Global analysis of patterns of gene expression during Drosophila embryogenesis.," *Genome biology*, vol. 8, no. 7, p. R145, 2007.

[23] H. Peng, F. Long, J. Zhou, G. Leung, M. B. Eisen, and E. W. Myers, "Automatic image analysis for gene expression patterns of fly embryos.," *BMC cell biology*, vol. 8 Suppl 1, p. S7, 2007.

[24] D. A. Baker and S. Russell, "Gene expression during Drosophila melanogaster egg development before and after reproductive diapause," *BMC Genomics*, vol. 10, p. 242, 2009.

[25] I. Pruteanu-Malinici, D. L. Mace, and U. Ohler, "Automatic annotation of spatial expression patterns via sparse bayesian factor models," *PLoS Computational Biology*, vol. 7, no. 7, 2011.

[26] A. A. S. Hammonds, C. C. a. Bristow, W. W. Fisher, R. Weiszmann, S. Wu, V. Hartenstein, M. Kellis, B. Yu, E. Frise, and S. E. Celniker, "Spatial expression of transcription factors in Drosophila embryonic organ development.," *Genome biology*, vol. 14, no. 12, p. R140, 2013.

[27] T. Kazmar, E. Z. Kvon, A. Stark, and C. H. Lampert, "Drosophila embryo stage annotation using label propagation," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1089–1096, 2013.

[28] C. Ayyub, K. K. Banerjee, and P. Joti, "Reduction of Cullin-2 in somatic cells disrupts differentiation of germline stem cells in the Drosophila ovary," *Developmental Biology*, vol. 405, no. 2, pp. 269–279, 2015.

[29] L. Gilboa, "Organizing stem cell units in the Drosophila ovary," *Current Opinion in Genetics and Development*, vol. 32, pp. 31–36, 2015.

[30] D. L. Brower, M. Wilcox, M. Piovant, R. J. Smith, and L. a. Reger, "Related cell-surface antigens expressed with positional specificity in Drosophila imaginal discs.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 81, no. 23, pp. 7485–9, 1984.

[31] D. L. Brower, "Engrailed gene expression in Drosophila imaginal discs.," *The EMBO journal*, vol. 5, no. 10, pp. 2649–2656, 1986.

[32] W. Li, J. T. Ohlmeyer, M. E. Lane, and D. Kalderon, "Function of protein kinase A in hedgehog signal transduction and Drosophila imaginai disc development," *Cell*, vol. 80, no. 4, pp. 553–562, 1995.

[33] P. Ahammad, C. L. Harmon, A. Hammonds, S. S. Sastry, and G. M. Rubin, "Joint Nonparametric Alignment for Analizing Spatial Gene Expression Patterns in Drosophila Imaginal Discs," in *CVPR*, vol. 2, 2005.

[34] G. Schubiger, M. Schubiger, and A. Sustar, "The three leg imaginal discs of Drosophila: "Vive la différence"," *Developmental Biology*, vol. 369, no. 1, pp. 76–90, 2012.

[35] F. Marty, C. Rockel-Bauer, N. Simigdala, E. Brunner, and K. Basler, "Large-scale imaginal disc sorting: A protocol for "omics"-approaches," *Methods*, vol. 68, no. 1, pp. 260–264, 2014.

[36] R. M. Parton, A. M. Vallés, I. M. Dobbie, and I. Davis, "Isolation of Drosophila egg chambers for imaging," *Cold Spring Harbor protocols*, vol. 2010, p. pdb.prot5402, apr 2010.

[37] F. Jug, T. Pietzsch, S. Preibisch, and P. Tomancak, "Bioimage Informatics in the context of Drosophila research," *Methods*, vol. 68, no. 1, pp. 60–73, 2014.

[38] J. Borovec, J. Kybic, and R. Nava, "Detection and Localization of Drosophila Egg Chambers in Microscopy Images," in *Machine Learning in Medical Imaging: 8th International Workshop, MLMI 2017* (Q. Wang, Y. Shi, H.-I. Suk, and K. Suzuki, eds.), (Cham), pp. 19–26, Springer International Publishing, 2017.

[39] J. Borovec, J. Kybic, and A. Sugimoto, "Region growing using superpixels with learned shape prior," *Journal of Electronic Imaging*, vol. 26, no. 6, pp. 26 – 26 – 14, 2017.

[40] J. Borovec, J. Svihlik, J. Kybic, and D. Habart, "Supervised and unsupervised segmentation using superpixels, model estimation, and Graph Cut," *Journal of Electronic Imaging*, vol. 26, no. 6, pp. 26 – 26 – 17, 2017.

[41] D. Jia, Q. Xu, Q. Xie, W. Mio, and W.-M. Deng, "Automatic stage identification of Drosophila egg chamber based on DAPI images," *Scientific Reports*, vol. 6, no. November 2015, p. 18850, 2016.

[42] J. Kybic and J. Borovec, "Automatic simultaneous segmentation and fast registration of histological images," in *International Symposium on Biomedical Imaging, IEEE*, pp. 774 – 777, 2014.

[43] J. Kybic, M. Dolejsi, and J. Borovec, "Fast registration of segmented images by normal sampling," in *Bio Image Computing (BIC) workshop at CVPR*, pp. 11–19, 2015.

[44] J. Borovec and J. Kybic, "Binary pattern dictionary learning for gene expression representation in drosophila imaginal discs," in *Mathematical and Computational Methods in Biomedical Imaging and Image Analysis (MCBMIIA) workshop at ACCV*, pp. 555–569, Springer, 2016.

[45] D. L. Pham, C. Xu, and J. L. Prince, "A Survey of Current Methods in Medical Image Segmentation," *In Annual Review of Biomedical Engineering*, vol. 2, pp. 315–338, 2000.

[46] K.-P. Wong, "Medical Image Segmentation: Methods and Applications in Functional Imaging," in *Handbook of Biomedical Image Analysis* (J. S. Suri, D. L. Wilson, and S. Laxminarayan, eds.), pp. 111–182, Springer US, 2005.

[47] A. Elnakib, G. Gimel'farb, J. Suri, and A. El-Baz, "Medical Image Segmentation: A Brief Survey," in *Multi Modality State-of-the-Art Medical Image Segmentation and Registration Methodologies* (A. S. El-Baz, R. Acharya U, A. F. Laine, and J. S. Suri, eds.), pp. 1–39, Springer New York, 2011.

[48] M. Hall, E. Frank, and G. Holmes, "The WEKA data mining software: an update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," pp. 1–8, 2015.

[50] F. Milletari, N. Navab, and S.-a. Ahmadi, "V-Net : Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation V-Net : Fully Convolutional Neural Networks for," no. September, pp. 1–11, 2016.

[51] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An Evaluation of the State-of-the-Art," *Computer Vision and Image Understanding*, vol. abs/1612.0, no. 35, pp. –, 2016.

[52] B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9910 LNCS, pp. 312–329, 2016.

[53] M. Ren and R. S. Zemel, "End-to-End Instance Segmentation with Recurrent Attention," *1605.09410v2*, pp. 1–17, 2016.

[54] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," 2017.

[55] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, 2014.

[56] R. Nava and J. Kybic, "Supertexton-based segmentation in early Drosophila oogenesis," in *Proceedings - International Conference on Image Processing, ICIP*, vol. 2015-Decem, pp. 2656–2659, 2015.

[57] Q. C. Q. Chen, X. Y. X. Yang, and E. Petriu, "Watershed segmentation for binary images with different distance transforms," in *Proceedings. Second International Conference on Creating, Connecting and Collaborating through Computing*, vol. 2, pp. 111–116, 2004.

[58] C. Tang, L. Zhang, A. Zhang, and M. Ramanathan, "Interrelated two-way clustering: An unsupervised approach for gene expression data analysis," in *International Symposium on Bioinformatics and Bioengineering*, pp. 41–48, 2001.

[59] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 11, pp. 1370–1386, 2004.

[60] A. Prelic, S. Bleuler, P. Zimmermann, A. Wille, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler, "A systematic comparison and evaluation of biclustering methods for gene expression data," *Bioinformatics*, vol. 22, no. 9, pp. 1122–1129, 2006.

[61] R. Simon, A. Lam, M. C. Li, M. Ngan, S. Menenzes, and Y. Zhao, "Analysis of gene expression data using BRB-ArrayTools," *Cancer Inform*, vol. 3, pp. 11–17, 2007.

[62] H. Y. Zhao, A. W. C. Liew, D. Z. Wang, and H. Yan, "Biclustering Analysis for Pattern Discovery: Current Techniques, Comparative Studies and Applications," *Current Bioinformatics*, vol. 7, pp. 43—-55, 2012.

[63] Q. Gao, C. Ho, Y. Jia, J. J. Li, and H. Huang, "Biclustering of Linear Patterns In Gene Expression Data," *Journal of Computational Biology*, vol. 19, no. 6, pp. 619–631, 2012.

[64] H. Ben Saber and M. Elloumi, "A Comparative Study of Clustering and Biclustering of Microarray Data," *International Journal of Computer Science and Information Technology*, vol. 6, no. 6, pp. 93–111, 2014.

[65] J. Klema, F. Malinka, and F. Zelezny, "Semantic biclustering: a new way to analyze and interpret gene expression data," in *Bioinformatics Research and Applications*, (Minsk, Belarus), pp. 332–3, Springer, 2016.

[66] J. Kim, K. Kim, and J. H. Kim, "Semantic Signature: Comparative Interpretation of Gene Expression on a Semantic Space," *Computational and Mathematical Methods in Medicine*, vol. 2016, pp. 1–10, 2016.

[67] S. Tweedie, M. Ashburner, K. Falls, and E. Al., "FlyBase: Enhancing Drosophila Gene Ontology annotations," *Nucleic Acids Research*, vol. 37, no. SUPPL. 1, pp. 555–559, 2009.

[68] J. Borovec and J. Kybic, "jSLIC : superpixels in ImageJ," in *Computer Vision Winter Workshop* (Z. Kukelova and J. Heller, eds.), (Praha), pp. 14–18, Czech Society for Cybernetics and Informatics, 2014.

[69] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. Cengage Learning, 3 ed., 2007.

[70] P. Neubert and P. Protzel, "Superpixel Benchmark and Comparison," *Tu-Chemnitz.De*, pp. 1–12, 2012.

[71] K. Puniyani, C. Faloutsos, and E. P. Xing, "SPEX2: Automated concise extraction of spatial gene expression patterns from fly embryo ISH images," *Bioinformatics*, vol. 26, no. 12, pp. 47–56, 2010.

[72] A. Lucchi, K. Smith, and R. Achanta, "Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks With Learned Shape Features," *Medical Imaging, IEEE*, vol. 31, no. 2, pp. 474 – 486, 2012.

[73] J. Borovec and J. Kybic, "Fully automatic segmentation of stained histological cuts," in *17th International Student Conference on Electrical Engineering* (L. Husník, ed.), (Prague), pp. 1–7, CTU in Prague, 2013.

[74] L. Lalaoui, T. Mohamadi, and A. Djaalab, "New Method for Image Segmentation," *Procedia - Social and Behavioral Sciences*, vol. 195, pp. 1971–1980, 2015.

[75] M. E. A. Bechar, N. Settouti, V. Barra, and M. A. Chikh, "Semi-supervised superpixel classification for medical images segmentation: application to detection of glaucoma disease," *Multidimensional Systems and Signal Processing*, vol. 28, no. 97, pp. 1–20, 2017.

[76] D. Boschetto and E. Grisan, "Superpixel-based classification of gastric chromoendoscopy images," in *Medical Imaging, SPIE*, vol. 10134, p. 101340W, 2017.

[77] O. Csillik, "Fast Segmentation and Classification of Very High Resolution Remote Sensing Data Using," *Remote Sensing*, vol. 9, no. 243, p. 19, 2017.

[78] Y. Boykov, "Graph cuts and efficient nd image segmentation," *International Journal of Computer Vision*, vol. 70, pp. 109–131, nov 2006.

[79] A. Lucchi, K. Smith, R. Achanta, and V. Lepetit, "A fully automated approach to segmentation of irregularly shaped cellular structures in EM images," in *Medical Image Computing and Computer-Assisted Intervention*, pp. 463–471, 2010.

[80] T. Kitrungrotsakul, X.-H. Han, and Y.-W. Chen, "Liver segmentation using superpixel-based graph cuts and restricted regions of shape constraints," in *International Conference on Image Processing (ICIP)*, vol. 3, pp. 3368–3371, 2015.

[81] J. Borovec, J. Kybic, M. Bušta, C. Ortiz-de Solorzano, and A. Munoz-Barrutia, "Registration of multiple stained histological sections," in *International Symposium on Biomedical Imaging, IEEE*, (San Francisco), pp. 1034–1037, 2013.

[82] A. Szmul, B. W. Papie, R. Bates, A. Hallack, J. A. Schnabel, and V. Grau, "Graph Cuts-Based Registration Revisited : A Novel Approach for Lung Image Registration Using Supervoxels and Image-Guided Filtering," in *CVPR 2016*, pp. 152–159, 2016.

[83] J. Yan, Y. Yu, X. Zhu, Z. Lei, and S. Z. Li, "Object detection by labeling superpixels," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 5107–5116, 2015.

[84] L. Limited, "Multiple-organ Segmentation by Graph Cuts with Supervoxel Nodes," No. 1, pp. 2–5, 2017.

[85] S. He, "SuperCNN: A Superpixelwise Convolutional Neural Network for Salient Object Detection," pp. 330–344, 2015.

[86] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel Convolutional Networks using Bilateral Inceptions," vol. 3, pp. 1–17, 2015.

[87] R. Achanta and A. Shaji, "SLIC Superpixels Compared to State-of-the-art Superpixel Methods," *Pattern Analysis and Machine Intelligence, IEEE*, vol. 34, no. 11, pp. 2274 – 2282, 2012.

[88] V. Machairas, E. Decenciere, and T. Walter, "Waterpixels: Superpixels based on the watershed transformation," *2014 IEEE International Conference on Image Processing, ICIP 2014*, pp. 4343–4347, 2014.

[89] X. Ren and J. Malik, "Learning a classification model for segmentation," *Proceedings Ninth IEEE International Conference on Computer Vision*, vol. 1, no. c, pp. 10–17 vol.1, 2003.

[90] J. Hartigan and M. Wong, "Algorithm AS 136: A K-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, pp. 100–108, oct 1979.

[91] C. C. Y. Ren and I. Reid, "gSLIC: a real-time implementation of SLIC superpixel segmentation," tech. rep., 2011.

[92] R. Birkus and I. Wanda, "Accelerated gSLIC for Superpixel Generation used in Object Segmentation," in *Proceedings of CESCG*, pp. 27–30, 2015.

[93] Z. Ban, J. Liu, and J. Fouriaux, "GLSC: LSC superpixels at over 130 FPS," *Journal of Real-Time Image Processing*, pp. 1–12, 2016.

[94] E. B. Alexandre, A. S. Chowdhury, A. X. Falcao, and P. A. Miranda, "IFT-SLIC: A General Framework for Superpixel Generation Based on Simple Linear Iterative Clustering and Image Foresting Transform," *Brazilian Symposium of Computer Graphic and Image Processing*, vol. 2015-Octob, pp. 337–344, 2015.

[95] F. Kou, Z. Li, C. Wen, and W. Chen, "Variance adaptive SLIC," *Proceedings of the 2016 IEEE 11th Conference on Industrial Electronics and Applications, ICIEA 2016*, pp. 1671–1675, 2016.

[96] G. Xuan and W. Zhang, "EM algorithms of Gaussian mixture model and hidden Markov model," *Image Processing, 2001.*, vol. 1, pp. 145–148, 2001.

[97] A. Tremeau and P. Colantoni, "Regions adjacency graph applied to color image segmentation," *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp. 735–744, 2000.

[98] Y. Boykov and O. Veksler, "Fast approximate energy minimization via graph cuts," *Pattern Analysis and Machine Intelligence, IEEE*, vol. 23, no. 11, pp. 1222–1239, 2001.

[99] F. Yi and I. Moon, "Image segmentation: A survey of graph-cut methods," in *2012 International Conference on Systems and Informatics, ICSAI 2012*, pp. 1936–1941, 2012.

[100] X. Ye, G. Beddoe, and G. Slabaugh, "Automatic graph cut segmentation of lesions in CT using mean shift superpixels," *International Journal of Biomedical Imaging*, 2010.

[101] C. Y. Hsu and J. J. Ding, "Efficient image segmentation algorithm using SLIC superpixels and boundary-focused region merging," in *ICICS 2013 - Conference Guide of the 9th International Conference on Information, Communications and Signal Processing*, 2013.

[102] X. Wang, H. Li, C.-E. Bichot, S. Masnou, and L. Chen, "A graph-cut approach to image segmentation using an affinity graph based on l0-sparse representation of features," in *2013 IEEE International Conference on Image Processing*, pp. 4019–4023, 2013.

[103] S. Zucker, "Region growing: Childhood and adolescence," *Computer Graphics and Image Processing*, vol. 21, pp. 269–399, 1976.

[104] D. Bailey, "Raster based region growing," in *Proceedings of the 6th New Zealand Image*, no. August, pp. 21–26, 1991.

[105] R. Adams and L. Bischof, "Seeded Region Growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641–647, 1994.

[106] C. Revol-Muller, T. Grenier, J. L. Rose, A. Pacureanu, F. Peyrin, and C. Odet, "Region Growing: When Simplicity Meets Theory - Region Growing Revisited in Feature Space and Variational Framework," in *Communications in Computer and Information Science*, vol. 359 CCIS, pp. 426–444, 2013.

[107] L. Vese and T. Chan, "A multiphase level set framework for image segmentation using the Mumford and Shah model," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 271–293, 2002.

[108] T. Chan and W. Zhu, "Level set based shape prior segmentation," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, pp. 1164–1170, 2005.

[109] O. Dzyubachyk, W. A. Van Cappellen, J. Essers, W. J. Niessen, and E. Meijering, "Advanced level-set-based cell tracking in time-lapse fluorescence microscopy," *IEEE Transactions on Medical Imaging*, vol. 29, no. 3, pp. 852–867, 2010.

[110] J. C. Russ, *The Image Processing Handbook*. 2002.

[111] H.-b. Tan, Z.-q. Hou, X.-c. Li, R. Liu, and W.-w. Guo, "Improved watershed algorithm for color image segmentation," *Proceedings of SPIE*, vol. 7495, no. 60805015, pp. 74952Z–74952Z–8, 2009.

[112] Q. Y. Q. Yu and D. Clausi, "IRGS: Image Segmentation Using Edge Penalties and Region Growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 12, pp. 2126–2139, 2008.

[113] A. K. Qin and D. A. Clausi, "Multivariate image segmentation using semantic region growing with adaptive edge penalty," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2157–2170, 2010.

[114] M. Kumar, P. Torr, and A. Zisserman, "Obj Cut," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 18–25, 2005.

[115] D. Freedman and T. Zhang, "Interactive graph cut based segmentation with shape priors," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. I, pp. 755–762, 2005.

[116] N. Vu and B. S. Manjunath, "Shape prior segmentation of multiple objects with graph cuts," in *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008.

[117] O. Veksler, "Star Shape Prior for Graph-Cut Image Segmentation," in *Computer Vision–ECCV*, pp. 454—-467, Springer, 2008.

[118] K. Nakagomi, A. Shimizu, H. Kobatake, M. Yakami, K. Fujimoto, and K. Togashi, "Multi-shape graph cuts with neighbor prior constraints and its application to lung segmentation from a chest CT volume," *Medical Image Analysis*, vol. 17, no. 1, pp. 62–77, 2013.

[119] T. Schoenemann and D. Cremers, "Globally optimal image segmentation with an elastic shape prior," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007.

[120] A. Delong and Y. Boykov, "Globally optimal segmentation of multi-region objects," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 285–292, 2009.

[121] J. Ulen, P. Strandmark, and F. Kahl, "An efficient optimization framework for multi-region segmentation based on lagrangian duality," *IEEE Transactions on Medical Imaging*, vol. 32, no. 2, pp. 178–188, 2013.

[122] H. Isack, O. Veksler, I. Iguz, M. Sonka, and Y. Boykov, "Efficient optimization for hierarchically-structured Interacting Segments (HINTS)," pp. 1445–1453, 2017.

[123] H. Lu, Y. Li, Y. Wang, S. Serikawa, B. Chen, and C. Chang, "Active Contours Model for Image Segmentation: A Review," in *The Proceedings of the 1st International Conference on Industrial Application Engineering 2013*, pp. 104–111, 2013.

[124] K. Zhang, L. Zhang, H. Song, and W. Zhou, "Active contours with selective local or global segmentation: A new formulation and level set method," *Image and Vision Computing*, vol. 28, no. 4, pp. 668–676, 2010.

[125] B. C. Lucas, M. Kazhdan, and R. H. Taylor, "Multi-Object Geodesic Active Contours (MOGAC)," *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012*, vol. 7511, pp. 404–412, 2012.

[126] N. Paragiosl and R. Deriche, "Coupled Geodesic Active Regions for Image Segmentation: A Level Set Approach," *Computer Vision*, pp. 224–240, 2000.

[127] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active Shape Models-Their Training and Application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[128] M. Leventon, W. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition CVPR*, 2000.

[129] A. Tsai, A. Y. Jr, and W. Wells, "A shape-based approach to the segmentation of medical imagery using level sets," *Medical Imaging,*, 2003.

[130] M. Gastaud, M. Barlaud, and G. Aubert, "Combining Geometric Prior And Statistical Features For Active Contour Segmentation," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. may, pp. 2–10, 2003.

[131] C. Molnar, Z. Kato, and I. Jermyn, "A Multi-Layer Phase Field Model for Extracting Multiple Near-Circular," in *International Conference on Pattern Recognition*, no. ICPR, pp. 1427–1430, 2012.

[132] C. Molnar, I. H. Jermyn, Z. Kato, V. Rahkama, P. Östling, P. Mikkonen, V. Pietiäinen, and P. Horvath, "Accurate Morphology Preserving Segmentation of Overlapping Cells based on Active Contours," *Scientific Reports*, vol. 6, no. 1, pp. 1–10, 2016.

[133] Y. Shi and W. C. Karl, "Real-time Tracking Using Level Sets," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, no. c, pp. 20–25, 2005.

[134] J. Kybic and J. Krátký, "Discrete curvature calculation for fast level set segmentation," *Proceedings - International Conference on Image Processing, ICIP*, pp. 3017–3020, 2009.

[135] P. Buyssens, I. Gardin, S. Ruan, and A. Elmoataz, "Eikonal-based region growing for efficient clustering," *Image and Vision Computing*, vol. 32, no. 12, pp. 1045–1054, 2014.

[136] K. Smith and A. Carleton, "Fast ray features for learning irregular shapes," in *Computer Vision*, pp. 397 – 404, 2009.

[137] K.-m. Lee and W. N. Street, "Learning shapes for automatic image segmentation," in *Proc. INFORMS-KORMS Conference*, pp. 1461–1468, 2000.

[138] J. L. Rose, C. Revol-Muller, M. Almajdub, E. Chereul, and C. Odet, "Shape prior integrated in an automated 3D region growing method," in *Proceedings - International Conference on Image Processing, ICIP*, vol. 1, 2007.

[139] J. L. Rose, C. Revol-Muller, J. B. Langlois, M. Janier, and C. Odet, "3D region growing integrating adaptive shape prior," in *2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Proceedings, ISBI*, pp. 967–970, 2008.

[140] J. L. Rose, C. Revol-Muller, D. Charpigny, and C. Odet, "Shape prior criterion based on tchebichef moments in variational region growing," in *Proceedings - International Conference on Image Processing, ICIP*, pp. 1081–1084, 2009.

[141] A. Quispe and C. Petitjean, "Shape prior based image segmentation using manifold learning," in *5th International Conference on Image Processing, Theory, Tools and Applications 2015, IPTA 2015*, pp. 137–142, 2015.

[142] P. Etyngier, F. Segonne, and R. Keriven, "Shape Priors using Manifold Learning Techniques," in *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, 2007.

[143] O. Moolan-Feroze, M. Mirmehdi, M. Hamilton, and C. Bucciarelli-Ducci, "Segmentation of the right ventricle using diffusion maps and Markov random fields," in *Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, vol. 8673 LNCS, pp. 682–689, 2014.

[144] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *ACM SIGGRAPH 2004 Papers on - SIGGRAPH '04*, p. 303, 2004.

[145] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Structured sparsity through convex optimization," pp. 1–27, 2011.

[146] H. Zou, T. Hastie, R. Tibshirani, I. Johnstone, and A. Lu, "Sparse principal component analysis," *Journal of Computational and Graphical Statistics*, vol. 15, no. 2, pp. 1–29, 2006.

[147] A. Hyvarinen, "Fast and Robust Fixed-Point Algorithm for Independent Component Analysis," *IEEE Trans. Neur. Net.*, vol. 10, no. 3, pp. 626–634, 1999.

[148] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pp. 1–8, 2009.

[149] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization.," *Neural computation*, vol. 19, pp. 2756–2779, 2007.

[150] S. Wu, A. Joseph, A. S. Hammonds, S. E. Celniker, B. Yu, and E. Frise, "Stability-driven nonnegative matrix factorization to interpret spatial gene expression and build local gene networks," *Proceedings of the National Academy of Sciences*, vol. 113, no. 16, p. 201521171, 2016.

[151] R. Belohlavek and V. Vychodil, "Discovery of optimal factors in binary data via a novel method of matrix decomposition," *Journal of Computer and System Sciences*, vol. 76, no. 1, pp. 3–20, 2010.

[152] Z. Y. Zhang, T. Li, C. Ding, X. W. Ren, and X. S. Zhang, "Binary matrix factorization for analyzing gene expression data," *Data Mining and Knowledge Discovery*, vol. 20, no. 1, pp. 28–52, 2010.

[153] A. Gersho and R. M. Gray, "Vector Quantization and Signal Compression," *Kluwer Academic Press*, vol. 159, p. 760, 1992.

[154] S. Lee, J. Z. Huang, and J. Hu, "Sparse logistic principal components analysis for binary data," *Annals of Applied Statistics*, vol. 4, no. 3, pp. 1579–1601, 2010.

[155] Y. Shen and G. B. Giannakis, "Online dictionary learning from large-scale binary data," *24th European Signal Processing Conference, EUSIPCO 2016*, pp. 1808–1812, 2016.

[156] G. Varoquaux, A. Gramfort, F. Pedregosa, V. Michel, and B. Thirion, "Multi-subject dictionary learning to segment an atlas of brain spontaneous activity," *Information Processing in Medical Imaging*, vol. 6801 LNCS, pp. 562–573, 2011.

[157] A. Abraham, E. Dohmatob, B. Thirion, D. Samaras, and G. Varoquaux, "Extracting brain regions from rest fMRI with total-variation constrained dictionary learning," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8150 LNCS, pp. 607–615, 2013.

[158] H. Peng and E. W. Myers, "Comparing in situ mRNA expression patterns of drosophila embryos," *RECOMB '04: Proceedings of the eighth annual international conference on Resaerch in computational molecular biology*, pp. 157–166, 2004.

[159] O. O. Koyejo, N. Natarajan, P. K. Ravikumar, and I. S. Dhillon, "Consistent Multilabel Classification," *Advances in Neural Information Processing Systems*, pp. 3303–3311, 2015.

[160] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.

[161] R. Achanta, A. Shaji, K. Smith, and A. Lucchi, "Slic superpixels," tech. rep., 2010.

[162] A. Vedaldi and B. Fulkerson, "VLFeat - An open and portable library of computer vision algorithms," in *Proceedings of the international conference on Multimedia - MM '10*, p. 1469, 2010.

[163] D. Martin and C. Fowlkes, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *International Conference on Computer Vision, IEEE*, no. July, 2001.

[164] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International journal of computer vision*, vol. 43, no. 1, pp. 29–44, 2001.

[165] A. Li, X. Wang, K. Yan, C. Li, and D. Feng, "Multilevel affinity graph for unsupervised image segmentation," in *ICIP*, 2016.

[166] C. A. Schneider, W. S. Rasband, and K. W. Eliceiri, "NIH Image to ImageJ: 25 years of image analysis," *Nature Methods*, vol. 9, pp. 671–675, jun 2012.

[167] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[168] I. Arganda-Carreras, V. Kaynig, C. Rueden, K. W. Eliceiri, J. Schindelin, A. Cardona, and H. Sebastian Seung, "Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification," *Bioinformatics*, vol. 33, no. 15, p. btx180, 2017.

[169] S. Bagon, "Matlab Wrapper for Graph Cut," 2006.

[170] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[171] R. Hal and J. Flusser, "Numerically stable direct least squares fitting of ellipses," *Proc. 6th International Conference in Central Europe on Computer Graphics and Visualization*, vol. 98, no. WSCG, pp. 125–132, 1998.

[172] S. Beucher, "The Watershed Transformation Applied to Image Segmentation," in *Proceedings of the 10th Pfefferkorn Conference on Signal and Image Processing in Microscopy and Microanalysis*, no. March, pp. 299–314, 1992.

[173] X. Ji, Y. Li, J. Cheng, Y. Yu, and M. Wang, "Cell image segmentation based on an improved watershed algorithm," in *Proceedings - 2015 8th International Congress on Image and Signal Processing, CISP 2015*, pp. 433–437, 2016.

[174] P. Marquez-Neila, L. Baumela, and L. Alvarez, "A morphological approach to curvature-based evolution of curves and surfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 2–17, 2014.

[175] G. Varoquaux, S. Sadaghiani, P. Pinel, a. Kleinschmidt, J. B. Poline, and B. Thirion, "A group model for stable multi-subject ICA on fMRI datasets," *NeuroImage*, vol. 51, no. 1, pp. 288–299, 2010.

A

Author's publications

A.1 Publications related to the Thesis

Journal publications

[40] J. Borovec, J. Svihlik, J. Kybic, and D. Habart, "Supervised and unsupervised segmentation using superpixels, model estimation, and Graph Cut," *Journal of Electronic Imaging*, vol. 26, no. 6, pp. 26 – 26 – 17, 2017. *[Authorship 70%]*

[39] J. Borovec, J. Kybic, and A. Sugimoto, "Region growing using superpixels with learned shape prior," *Journal of Electronic Imaging*, vol. 26, no. 6, pp. 26 – 26 – 14, 2017. *[Authorship 70%]*

Workshop publications

[73] J. Borovec and J. Kybic, "Fully automatic segmentation of stained histological cuts," in *17th International Student Conference on Electrical Engineering* (L. Husník, ed.), (Prague), pp. 1–7, CTU in Prague, 2013. *[Authorship 90%]*

[68] J. Borovec and J. Kybic, "jSLIC : superpixels in ImageJ," in *Computer Vision Winter Workshop* (Z. Kukelova and J. Heller, eds.), (Praha), pp. 14–18, Czech Society for Cybernetics and Informatics, 2014. *[Authorship 90%]*

[44] J. Borovec and J. Kybic, "Binary pattern dictionary learning for gene expression representation in drosophila imaginal discs," in *Mathematical and Computational Methods in Biomedical Imaging and Image Analysis (MCBMIIA) workshop at ACCV*, pp. 555–569, Springer, 2016. *[Authorship 70%]*

[38] J. Borovec, J. Kybic, and R. Nava, "Detection and Localization of Drosophila Egg Chambers in Microscopy Images," in *Machine Learning in Medical Imaging: 8th International Workshop, MLMI 2017* (Q. Wang, Y. Shi, H.-I. Suk, and K. Suzuki, eds.), (Cham), pp. 19–26, Springer International Publishing, 2017. *[Authorship 70%]*

A.2 Publications unrelated to this thesis

[81] J. Borovec, J. Kybic, M. Bušta, C. Ortiz-de Solorzano, and A. Munoz-Barrutia, "Registration of multiple stained histological sections," in *International Symposium on Biomedical Imaging, IEEE*, (San Francisco), pp. 1034–1037, 2013

[42] J. Kybic and J. Borovec, "Automatic simultaneous segmentation and fast registration of histological images," in *International Symposium on Biomedical Imaging, IEEE*, pp. 774 – 777, 2014

[43] J. Kybic, M. Dolejsi, and J. Borovec, "Fast registration of segmented images by normal sampling," in *Bio Image Computing (BIC) workshop at CVPR*, pp. 11–19, 2015