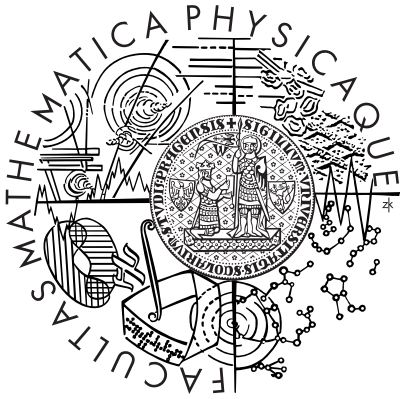
Charles University in Prague

Faculty of Mathematics and Physics

MASTER THESIS



Zuzana Bílková

Segmentation of Microscopic Images Using Level Set Methods

Department of Numerical Mathematics

Supervisor of the thesis: RNDr. Václav Kučera, Ph.D.

Consultant: RNDr. Jindřich Soukup

Study programme: Mathematics

Specialization: : Numerical and Computational Mathematics

Prague 2015

Acknowledgement

I would like to thank my supervisor RNDr. Václav Kučera, Ph.D. and my consultant RNDr. Jindřich Soukup for their time dedicated to consultations and reading this master thesis, for many valuable suggestions and comments.

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In Prague, 30. 7. 2015

Zuzana Bílková

Název práce: Segmentace mikroskopických snímků pomocí level-set metod

Autor: Zuzana Bílková

Katedra: Katedra numerické matematiky

Vedoucí diplomové práce: RNDr. Václav Kučera, Ph.D., KNM, MFF UK

Konzultant: RNDr. Jindřich Soukup, ÚTIA, AV ČR

Abstrakt:

Tato diplomová práce představuje novou metodu pro segmentaci snímků pořízených mikroskopem s fázovým kontrastem. Cílem je oddělit buňky od pozadí.

Algoritmus je založen na variační formulaci level set metod, tedy na minimalizaci funkcionálu popisujícího level set funkci. Funkcionál je minimalizován gradientním tokem popsaným evoluční parciální diferenciální rovnicí.

Nejdůležitější nové myšlenky jsou inicializace pomocí prahování a nové členy ve funkcionálu, které zrychlují konvergenci a zpřesňují výsledky. Také jsme použili nové funkce napsané v jazyce C k počítání gradientu a Laplaceova operátoru. Tato implementace je třikrát rychlejší než standardní funkce v MATLABu.

Dosáhli jsme lepších výsledků než algoritmy, se kterými jsme metodu porovnávali.

Klíčová slova: Segmentace, level set metody, aktivní kontury

Title: Segmentation of microscopic images using level set methods

Author: Zuzana Bílková

Department: Department of Numerical Mathematics

Supervisor: RNDr. Václav Kučera, Ph.D., Dept. of Numerical Mathematics

Consultant: RNDr. Jindřich Soukup, IITA, ASCR

Abstract:

This master thesis presents a new method for segmentation of phase-contrast microscopic images of cells. The goal is to segment the cells from the background.

The algorithm is based on the variational formulation of the level set method, i.e. minimizing of a functional, which describes the level set function. The functional is minimized by a gradient flow described by an evolutionary partial differential equation.

The most significant new ideas are initialization using thresholding and introducing new terms that speed up the convergence and achieve more accurate results. Moreover, we speed up the evaluation of gradient and Laplace operator using new functions written in C language. The new implementation is three times faster than the standard functions in MATLAB.

We compared the results with other algorithms and we achieved better accuracy.

Keywords: Segmentation, level set methods, active contours

Contents

1 Introduction

Image segmentation is one of the most important parts of *digital image processing*. It entails automatic division of an image into regions with similar attributes such as colour, intensity or texture. The typical goal of image segmentation is the identification of background and objects in the foreground.

Image segmentation is applicable in many disciplines, e.g. machine vision, object detection, astronomy or recognition tasks, such as face recognition, fingerprint recognition or license plates recognition [10, 17]. Another important discipline of image segmentation applications is *medicine*. Medical imaging is used e.g. for automatic locating of tumors and other pathologies, surgery planning or intra-surgery navigation.

The images we process are *microscopic images of cells*. Live cell imaging captures crucial information of many biological processes with direct implication for human health [16, 9, 14, 7]. The analysis of our images is used for the development of *body implants*, see [19]. The human body is very sensitive to foreign materials, unsuitable implants may cause immune reactions. Therefore, the biocompatibility or biotoxicity of various materials is studied.

Current methods of image segmentation are developed based on laboratory experiments in vitro which are not as expensive or time consuming as the testing of materials in clinical studies. The images we process come from experiments in vitro using *cancer cells* due to their resistance and easy laboratory preservation. It is assumed that if the cancer cells are not able to survive in the testing environment, then neither the normal body cells can survive.

The cells are scanned with a microscope at regular time intervals, in our case every 2 minutes. The images are analyzed to determine the rate of cell growth, i.e. this leads to the problem of *segmentation of cells from the background*. The evolution of area covered by cells describes the desired rate of cell growth.

The main goal is to find an efficient algorithm since the segmentation is usually performed manually which is a tedious and time consuming work.

Automatic segmentation is a complicated problem in itself and in our case there are several factors that make our task even more difficult. The microscopic images contain artifacts like halos, bright areas around the cell borders. The colour of the background is inconveniently similar to the colour of the cell interiors. The microscopic images also suffer from poor focus and impurities.

There are various segmentation methods in the literature cf. [21] for an overview. In this thesis, we focus on the method of *active contours* and its *level set formulation.* We use the level set method introduced by Osher [13]. The main idea behind the level set method is that a curve can be seen as the implicitly given zero level set of a function in higher dimension. The goal is to capture and analyze the *motion* of the curve during the calculation. Instead of moving the curve itself, we will be moving the level set function.

The proposed algorithm is based on the variational formulation of the level set method, i.e. minimizing of a functional. The functional describes the interface separating the cell clusters from the background. The interface is given implicitly by the level set function. The functional is minimized by an evolutionary partial differential equation describing its gradient flow. The equation is then discretized with the finite difference method.

We present a functional proposed in [8] and we incorporate new terms into its gradient flow that speed up the evolution and help to achieve more accurate results. The method is simple and versatile.

The following chapter gives mathematical background and description of our method. We discuss segmentation as a variational method and define evolution of explicit and implicit curves, as well as the level set method and the gradient flow formulation. Implementation and numerical scheme is explained in the third chapter. We also present different approaches of initialization and automatic optimization of weighting parameters in the third chapter. The last chapter entails numerical experiments. The results are presented and compared with another algorithm.

2 Method Description

In the following sections we derive the mathematical formulation of the level set method, first suggested in [13]. We will introduce our method and explain each term in our functional and its importance. We will assume that all functions and functionals satisfy conditions needed for the theoretical derivation of the method.

2.1 Mathematical Formulation

Digital Images

In the next sections, we will derive the method generally for functions defined in the following section and then we will apply it for the digital images. To keep up with the general formulation, we explain the digital images first.

An *image* u is a two dimensional function

$$u : \Omega \to \mathbb{R},$$

where $\Omega \subset \mathbb{R}^2$ is the image support.

A *digital image* is a numerical representation of a two dimensional image, i.e. it is a *discrete function*. The digital image is described by discrete points, called *pixels*. The pixels are arranged in a grid and each pixel has its position, represented by the space coordinates, and colour. The colour is also discretized and its values are natural numbers between 0 and 255. In the case of *grayscale images*, a pixel with the value 0 represents a black pixel, the pixel with the value 255 is a white pixel. *Coloured images* can be represented by RGB colour model, which is a triplet of intensity values of red, green and blue colours.

2.1.1 Segmentation As a Variational Method

Let X be a Banach space, $u \epsilon X$ and $F : X \to \mathbb{R}$. We will assume that X is a space of sufficiently regular functions. Calculus of variations then solves

$$\min_{u \in X} F(u), \tag{1}$$

which will play a key role in our method.

Let us denote the *partial derivative* of u with respect to the variable x_i as $\frac{\partial u}{\partial x_i}$ or u_{x_i} and let $\frac{du}{dx_i}$ be the *total derivative* with respect to x_i.

We will be using *integral functionals* in this thesis. In the case of images an example of the integral functional can be

$$F(u) = \int_{\Omega} f(x, u(x), \nabla u(x)) dx,$$

where $x \in \mathbb{R}^2$ are space coordinates (x_1, x_2), $\Omega \subset \mathbb{R}^2$ is the image support, $u : \Omega \to [0, 255]$ is a grayscale image (in case of a RGB image $u : \Omega \to [0, 255]^3$), $\nabla u = (u_{x_1}, u_{x_2})$ is the image gradient and f is a function that specifies our functional and is twice continuously differentiable.

To solve (1) we will use the *Euler-Lagrange equation*. For simplicity, we will derive the one-dimensional Euler-Lagrange equation in the next paragraphs. The two-dimensional version is derived analogically. We will use the following functional:

$$F(u) = \int_{a}^{b} f(x, u(x), u'(x)) dx,$$

where $a, b \in \mathbb{R}$, $a < b$, $u : [a, b] \to \mathbb{R}$, u' is its derivative and f is twice continuously differentiable.

Before deriving the Euler-Lagrange equation, some useful lemmas will be reminded first.

Lemma. Let $g : \mathbb{R}^m \to \mathbb{R}$, $h_i : \mathbb{R}^n \to \mathbb{R}$, for $i = 1, \ldots m$, $x \in \mathbb{R}^n$, $h(x) = (h_1(x), \ldots, h_m(x))$. The *chain rule* expresses a formula for computing the derivative of the composition $g(h(x))$:

$$\frac{\partial g}{\partial x_i} = \sum_{l=1}^{m} \frac{\partial g}{\partial h} \frac{\partial h_l}{\partial x_i}. \tag{2}$$

Lemma. Let $[a, b]$ be an interval and $g, h : [a, b] \to \mathbb{R}$. *Integration by parts* for functions g and h states:

$$\int_a^b (gh')\,dx = gh|_a^b - \int_a^b (g'h)\,dx, \tag{3}$$

where $gh|_a^b = g(b)h(b) - g(a)h(a)$.

Lemma. *Fundamental lemma of the calculus of variations:* Let g be a continuous function on the interval $[a, b]$. If

$$\int_a^b g(x)h(x)dx = 0 \tag{4}$$

for *every* function $h \in L^2[a, b]$, then

$$g(x) = 0, \quad \text{for all } x\epsilon(a, b).$$

Now we can derive the Euler-Lagrange equation. As mentioned above we seek an extremum, specifically the minimum, of the functional $F(u)$. If $u \in X$ is an extremum of F then from differential calculus it follows that it has to fulfil the necessary condition

$$\frac{d}{d\varepsilon}F(u + \varepsilon v)|_{\varepsilon=0} = 0 \quad \forall v,$$

where $v \in X$ and $\frac{d}{d\varepsilon}F(u+\varepsilon v)|_{\varepsilon=0}$ is the derivative of the functional F in the direction v. For

$$F(u + \varepsilon v) = \int_a^b f(x, u + \varepsilon v, u' + \varepsilon v')dx$$

we get

$$\frac{d}{d\varepsilon}F(u + \varepsilon v) = \frac{d}{d\varepsilon}\int_a^b f(x, u + \varepsilon v, u' + \varepsilon v')dx.$$

Because we assume $f \in C^2$ and $[a, b]$ is finite, then we can interchange the derivative and the integral:

$$\frac{d}{d\varepsilon}F(u+\varepsilon v)=\int_a^b \frac{d}{d\varepsilon}f(x, u+\varepsilon v, u'+\varepsilon v')dx.$$

Applying the chain rule (2) and omitting the arguments $(x, u+\varepsilon v, u'+\varepsilon v')$ for simplicity, we obtain

$$\frac{d}{d\varepsilon}F(u+\varepsilon v)=\int_a^b \left(\frac{\partial f}{\partial u}v+\frac{\partial f}{\partial u'}v'\right)dx.$$

Using integration by parts (3) on the last term on the right hand side of the equation we get

$$\begin{aligned}\frac{d}{d\varepsilon}F(u+\varepsilon v) &= \int_a^b \frac{\partial f}{\partial u}vdx-\int_a^b \frac{d}{dx}\left(\frac{\partial f}{\partial u'}\right)vdx+\frac{\partial f}{\partial u}v|_a^b \\ &= \int_a^b \left[\frac{\partial f}{\partial u}-\frac{d}{dx}\frac{\partial f}{\partial u'}\right]vdx+\frac{\partial f}{\partial u}v|_a^b.\end{aligned}$$

Since we assumed that $\frac{d}{d\varepsilon}F(u+\varepsilon v)|_{\epsilon=0}=0$, we get

$$\begin{aligned}\frac{d}{d\varepsilon}F(u+\varepsilon v)|_{\epsilon=0} &= \int_a^b \left[\frac{\partial f}{\partial u}-\frac{d}{dx}\frac{\partial f}{\partial u'}\right](x,u,u')vdx+ \\ &\quad +\frac{\partial f}{\partial u}v|_a^b=0 \quad \forall v\in X. \qquad (5)\end{aligned}$$

In order to enforce

$$\frac{\partial f}{\partial u}v|_a^b=0,$$

we will assume suitable boundary conditions, i.e. $v(a)=v(b)=0$. Therefore for (5) to be equal to 0 for any v satisfying the boundary conditions, we use the fundamental lemma of the calculus of variations (4) and we obtain the following equation:

$$\left[\frac{\partial f}{\partial u}-\frac{d}{dx}\frac{\partial f}{\partial u'}\right](x,u,u')=0. \qquad (6)$$

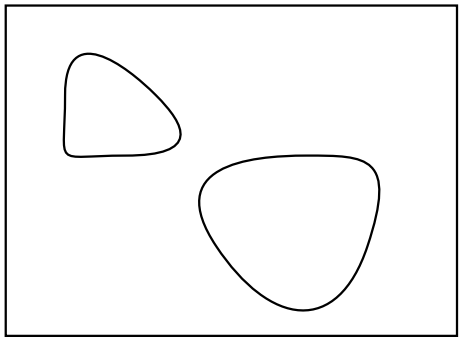


Figure 1: Interface - a set of curves dividing a plane into a background and a foreground.

The equation (6) is called the *Euler - Lagrange equation.*

Analogically to (6) we can now define the Euler-Lagrange equation in n dimensions.

Definition. If $u : \mathbb{R}^n \to \mathbb{R}$ is the extremum of $F(u)= \int_\Omega f(x, u(x), \nabla u(x))dx$, where $\nabla u \equiv (u_{x_1}, ..., u_{x_n})$, then

$$F'(u) = \frac{\partial f}{\partial u}(x, u, \nabla u) - \sum_{i=1}^{n} \frac{d}{dx_i} \left(\frac{\partial f}{\partial u_{x_i}}(x, u, \nabla u) \right) = 0, \tag{7}$$

which is the *Euler-Lagrange equation.*

2.1.2 Definition of Level Set Method

The *level set method* was first introduced by Osher and Sethian [13] as a simple and versatile method for numerical analysis of the motion of an interface.

An *interface* is a hyperplane, i.e. a subspace of dimension $n - 1$, which divides the space into two subspaces. In two dimensions, an interface is a finite set of closed curves, see (9), that divides the plane into the exterior and the interior, i.e. the background and the foreground. An interface in two dimensions is shown in Figure 1.

To describe the level set method we first define some terms. Let us denote the interface as c in $\mathbb{R}^n$ and the open region it is bounding as c^+ and the background as c^-.

Definition. Let $\phi : \mathbb{R}^n \to \mathbb{R}$ be a cotinuous function that is not identically zero on any open set. A *level set* of a function ϕ is a set where the values of the function are equal to a given constant *k:*

$$L_k(\phi) = \{(x_1, \ldots, x_n) | \phi(x_1, \ldots, x_n) = k\}.$$

A level set with the given constant of *0* is called a *zero level set*. In two dimensions a level set is generally a set of curves, called a *level curve*.

Level set methods use level sets as a tool for capturing the motion of interfaces.

Definition. The function ϕ from the previous Definition is called the *level set function*. We will assume that the level set function also depends on time, so $\phi : \mathbb{R}^{n+1} \to \mathbb{R}$, and denote the first variable as t and $x = (x_1, \ldots, x_n)$.

We will denote the *interior* and *exterior* of ϕ as ϕ^- and ϕ^+, respectively:

$$\begin{aligned} \phi^-(t) &= \{x \in \Omega : \phi(t, x) < 0\} \\ \phi^+(t) &= \{x \in \Omega : \phi(t, x) > 0\} \\ c(t) &= \{x \in \Omega : \phi(t, x) = 0\} \end{aligned} \tag{8}$$

In the case of images we assume $n = 2$, thus the level set c is a curve or set of curves, c^+ corresponds to ϕ^+ and c^- corresponds to ϕ^-. We will also consider only the zero level sets. In two dimensions, we will use the notation (x, y) instead of (x_1, x_2).

The *main idea* behind the level set method is that a curve can be seen as the implicitly given zero level set of a function in higher dimension, which is illustrated in Figure 2. The red curve in Figure 2 represents the zero level set.

The goal is to capture and analyze the *motion* of the curve c in time. Instead of moving the curve itself, we will be moving the level set function. This is the philosophy of level set methods.

The statement in the previous paragraph has great consequences for numerical computations. Primarily the *topological changes* of the curves represented by the level set function, such as merging or breaking, are performed naturally during the

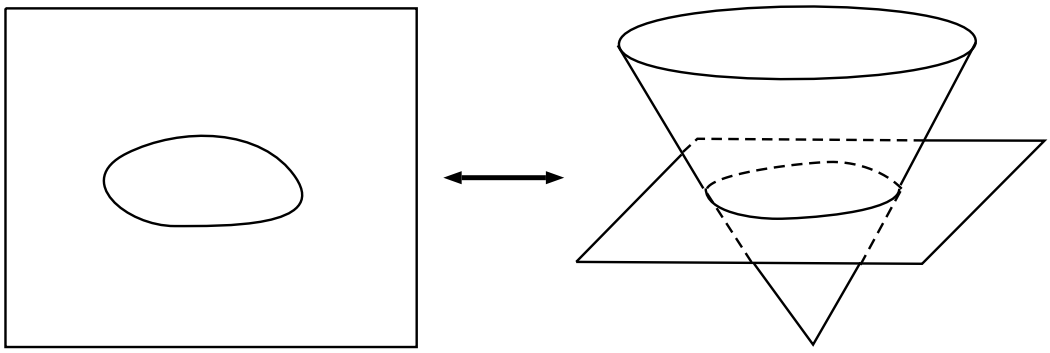


Figure 2: Illustration of the level set. The curve in the plane on the left side can be seen as a level set of a function in higher dimension shown on the right side.

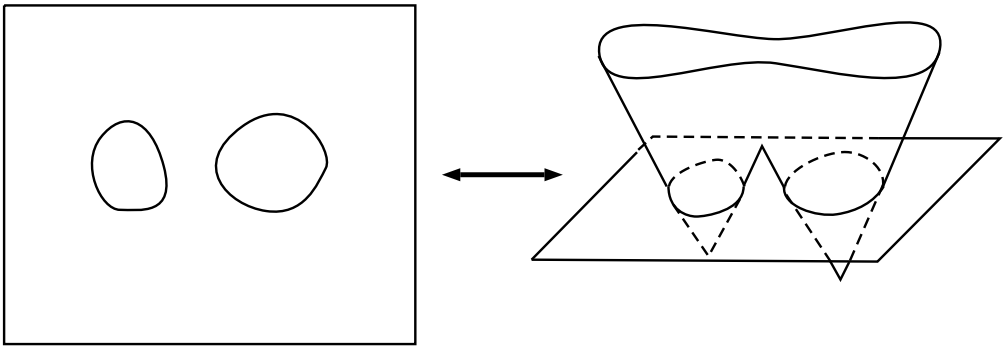


Figure 3: Illustration of topological changes of the interface during evolution of the level set function.

evolution as shown in the Figure 3. Second, the numerical approximation of a level set function stays on a fixed grid, which allows easy numerical approximation of differential equations governing the motion of c, e.g. finite-difference schemes. In addition, geometry elements, such as curvatures and normal vectors, can be directly computed from ϕ, as will be described later.

2.1.3 Evolution of Explicit and Implicit Curves

This section presents a brief introduction of *curve representation* and *curve evolution*. We will describe explicit parametric curves and curves given implicitly.

A time dependent *closed parametric curve*, denoted by c, is a continuous function

$$c : \mathbb{R}^+ \times [0,1] \to \Omega \subset \mathbb{R}^2,\ c(t,0) = c(t,1),$$
$$c(t,q) = [c_1(t,q), c_2(t,q)]. \tag{9}$$

Let us define some important geometrical terms: tangent and normal vectors and curvature.

The first derivative of the curve is called the *tangent vector*:

$$\mathbf{t} = c'(t,q) = \left(\frac{\partial c_1}{\partial q}(t,q), \frac{\partial c_2}{\partial q}(t,q) \right),$$

the derivative c', as in this whole paper, is the derivative with respect to q, similarly the second derivative c''.

The *unit normal vector*, omitting the arguments (t,q) for simplicity, is:

$$\mathbf{n} = \frac{(c')^\perp}{|c'|} = \frac{\left(-\frac{\partial c_2}{\partial q}, \frac{\partial c_1}{\partial q} \right)}{\sqrt{\left(\frac{\partial c_1}{\partial q} \right)^2 + \left(\frac{\partial c_2}{\partial q} \right)^2}},$$

where $(c')^\perp$ is *orthogonal* to c' and $|.|$ is the Euclidean norm.

Intuitively, the *curvature* κ tells how fast the curve deviates from the tangent direction in a given point. The curvature is the length of the normal direction component of the second derivative. The curvature is defined as, see [12],

$$\kappa = \|\mathbf{t}'\|. \tag{10}$$

We are interested in the motion of a curve over time and will consider evolution only in *normal direction* since it is geometrically relevant:

$$\frac{\partial c}{\partial t}(t,q) = F(t,q)\mathbf{n}(\mathbf{t},\mathbf{q}), \tag{11}$$

where F is a certain *speed function*. Such models describes e.g. crystal growth and F may depend e.g. on the curvature [12].

Another approach is to define a curve *implicitly*. At all times the curve is repre-

sented by the zero level set of a level set function $\phi(t, x, y)$:

$$\phi(t, c(t, q)) = 0. \tag{12}$$

In the implicit formulation normal and curvature take the following forms, see [12]:

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}$$

and the curvature is

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} = \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right), \tag{13}$$

where $\operatorname{div} v = \frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y}$, for $v : \mathbb{R}^2 \to \mathbb{R}$.

In the level set method, the function ϕ can be any arbitrary function as long as its zero level set matches the contour.

Differentiating the equation (12) with respect to t it follows that time derivative is zero,

$$0 = \frac{\partial}{\partial t}\phi(t, c(t, q)) = (\nabla\phi)\,(t, c(t, q)) \cdot \frac{\partial c}{\partial t}(t, q) + \frac{\partial \phi}{\partial t}(t, c(t, q)),$$

where we applied the chain rule (2) for the second equality. We then obtain an *evolution equation* for ϕ, omitting the arguments:

$$\frac{\partial\phi}{\partial t} = -\nabla\phi \cdot \frac{\partial c}{\partial t},$$

using (11) we get

$$\frac{\partial\phi}{\partial t} = -\nabla\phi \cdot F\mathbf{n}.$$

The equation (12) is satisfied on the curve $c(t, q)$, we reformulated it as an equation for ϕ and we will extend it for all x, t. Since $\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}$, the evolution equation can be written in the following general form [13]:

$$\frac{\partial\phi}{\partial t} = -F|\nabla\phi|,$$

where we omitted the arguments (t, x, y).

Definition. The equation

$$\frac{\partial\phi}{\partial t} = -F|\nabla\phi|$$

is called the *level set equation.*

The function F is called a *speed function.*

2.1.4 Variational Formulation of the Level Set Method

Early level set methods are usually represented as an evolutionary partial differential equation (PDE) of a parametrized curve in a Lagrangian framework converted to an evolutionary PDE for a level set function using an Eulerian framework. In this thesis, we present the evolution PDE as a variational problem [13, 12], thus minimizing an energy functional defined on the level set function.

The variational formulation of level set methods is more convenient for incorporating additional information about the desired solution. This property will be utilized later to eliminate the need of re-initialization procedure during implementation. Moreover, we will make use of the particular characteristics of the microscopic images of cells and introduce a new variance term in the gradient flow of the curve.

An explicit *energy functional* $E(\phi)$ will be defined, so that the zero level curve of the minimizer ϕ captures the desired features in an image, in our case the cell edges. The energy functional has two parts:

$$E(\phi) = E_m(\phi) + \mu P(\phi), \tag{14}$$

where $\mu > 0$ is a weighting parameter and ϕ does not depend on t. The energy $E_m(\phi)$ depends on the image data, therefore it is called the *external energy.* The energy $P(\phi)$ is a function of ϕ only, it is called the *internal energy.*

The internal energy $P(\phi)$ helps to produce more stable and robust results. In implementations of the traditional level set methods, it is numerically important to maintain the level set function close to a *signed distance function* [8]. The value of the signed distance function of a curve c in a given point $x \in \Omega$ determines the distance of x from c, with the sign defined by whether x is in the interior or exterior of c.

Definition. The *signed distance function* f of a set of curves c is defined by

$$f(x) = \begin{cases} d(x,c) & \text{for } x \in c^- \\ -d(x,c) & \text{for } x \in c^+ \end{cases},$$

where $d(x,c)$ is the distance of x from c.

From the definition, the signed distance function must satisfy

$$|\nabla f(x,t)| = 1 \quad \text{a.e. } x \in \Omega,\ \forall t. \tag{15}$$

This property of the signed distance function simplifies the formulas, e.g. the formula of curvature (13) is $\kappa = \operatorname{div}(\nabla\phi) = \triangle\phi$. Using the signed distance function also improves properties of the numerical approximation [1], e.g avoiding steep gradients.

During the evolution the level set function may develop irregularities, which can make further results highly inaccurate. Therefore, it is convenient for $\phi(t)$ to be a signed distance function for all t.

One approach is a periodical *re-initialization* of the level set function. Re-initialization constructs from ϕ the signed distance function $\tilde{\phi}$, which maintains the zero level set, $\{\phi = 0\}$. However, this technique has undesirable side-effects and it still remains a serious question of when and how to apply the re-initialization process. Li *et al.* [8] introduced a new internal energy term that penalizes the deviation of the level set function from the signed distance function property (15). Therefore the re-initialization is no longer necessary. In this thesis, we will use the following integral, proposed by them, as a *metric* to measure the distance between ϕ and a signed distance function:

$$P(\phi) = \int_\Omega \frac{1}{2}\left(|\nabla\phi| - 1\right)^2 dxdy. \tag{16}$$

We used the property (15). This metric will play a key role in our level set formulation.

The variational level set formulation proposed in [8] has also other advantages than re-initialization. In implementation of the gradient flow, defined later, a larger time step can be used comparing to the traditional formulation, which significantly speeds up the evolution. Furthermore, the level set function is no longer necessarily initialized as the signed distance function at $t = 0$. More general functions can be used that are easier to implement than the signed distance function. And finally, the

evolution can be implemented with a simple finite-difference scheme.

Our method uses the functional proposed in [8] and improves it for our problem. We first describe the functional which we want to minimize, then the *gradient flow* that minimizes it and finally we will present two terms, which we added to the gradient flow. One is our new *variance term*, the other one is a *mean value term*.

Before looking at the functional, some useful functions will be presented first.

Definition. The *Heaviside step function (Heaviside function)*:

$$H(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}. \tag{17}$$

The *Dirac delta distribution (delta distribution)* $\delta(x)$ is the distributive derivative of the Heaviside function, [5]. It satisfies

$$\int_{-\infty}^{\infty} \delta(x)dx = 1. \tag{18}$$

To achieve that the minimizer ϕ finds the object boundaries, we define the edge indicator function.

Definition. Let I be an image. We define the *edge indicator function* $g : \Omega \to \mathbb{R}$ as

$$g = \frac{1}{1 + |\nabla (G_\sigma * I)|^2},$$

where G_σ is the Gaussian kernel with standard deviation σ, $*$ means convolution.

Convolution of the image with the Gaussian kernel, $G_\sigma * I$, slightly blurs the image in order to eliminate noise which creates false edges. The size of the gradient of the blurred image, $|\nabla (G_\sigma * I)|$, is largest at the edges. Since we will use the edge indicator function in the minimization problem and we seek edges, we use the multiplicative inverse. To make sure we never divide by zero we add 1 to the denominator which also ensures that the values of g are between 0 and 1. The edge indicator function is thus minimal at the image edges.

We define the external energy as

$$E_m(\phi) = \lambda L(\phi) + \alpha A(\phi), \tag{19}$$

where $\lambda > 0$, α are constants and the terms $L(\phi)$ and $A(\phi)$ are defined as

$$L(\phi) \;=\; \int_\Omega g\delta(\phi)|\nabla\phi|dxdy \tag{20}$$

and

$$A(\phi) \;=\; \int_\Omega gH(-\phi)dxdy, \tag{21}$$

where δ is the delta distribution (18) and H is the Heaviside function (17).

It is well known [20] that the geometrical meaning of the energy $L(\phi)$ is the *length* of the zero level curve of the level set function ϕ in the conformal metric $ds = g(c(q))|c'(q)|dq$, where $c(q),\ q \in [0,1]$ is the zero level curve. Thus the functional $L(\phi)$ minimizes the length and therefore has a smoothing effect on the zero level curve.

The energy functional $A(\phi)$ minimizes the *area* of the region Ω^- and is proposed to speed up the curve evolution. If $g \equiv 1$, then $A(\phi)$ computes the area of Ω^-. The coefficient α in (19) can take positive or negative values, depending on the position of the initial contour to the object we want to detect. In our case the initial contour will be placed outside the cells, therefore the coefficient α should be positive, so that the contours shrink faster. Analogically, if the initial contour lays inside the object, α should take a negative value.

Definition. The *total energy functional* is defined by

$$\begin{aligned} E(\phi) &= \mu P(\phi) + E_m(\phi) \\ &= \mu \int_\Omega \frac{1}{2}\left(|\nabla\phi| - 1\right)^2 dxdy + \\ &\quad +\lambda \int_\Omega g\delta(\phi)|\nabla\phi|dxdy + \alpha \int_\Omega gH(-\phi)dxdy. \end{aligned} \tag{22}$$

2.1.5 Gradient flow formulation

We seek the *stationary solution* of the energy functional E given by (22), which is computed using the Euler-Lagrange equation (7). The *gradient flow* that minimizes E, is the following evolutionary equation:

$$\frac{\partial \phi}{\partial t} = -\frac{\partial E}{\partial \phi} = -E', \tag{23}$$

which is equivalent to solving the *variational problem* with the *steepest-descent method*.

To explicitly express the gradient flow we need to compute and express $\frac{\partial E}{\partial \phi}$. We apply the Euler-Lagrange equation (7) to the functional (22). The derivative is a linear operator, so we use the Euler-Lagrange equation separately for each term.

The penalizing term $P(\phi)$ is then

$$\begin{aligned} P(\phi) &= \int_\Omega \frac{1}{2}\left(|\nabla\phi| - 1\right)^2 dxdy \\ &= \frac{1}{2}\int_\Omega |\nabla\phi|^2 dxdy - \int_\Omega |\nabla\phi| dxdy + \int_\Omega \frac{1}{2} dxdy \\ &= P_1(\phi) - P_2(\phi) + \int_\Omega \frac{1}{2} dxdy, \end{aligned} \tag{24}$$

where $P_1(\phi) = \frac{1}{2}\int_\Omega |\nabla\phi|^2 dxdy$ and $P_2(\phi) = \int_\Omega |\nabla\phi| dxdy$. Using the following

$$|\nabla\phi| = \sqrt{\phi_x^2 + \phi_y^2}$$

and denoting $f\left(x, \phi, \nabla\phi\right) = \frac{1}{2}|\nabla\phi|^2 = \frac{1}{2}\left(\phi_x^2 + \phi_y^2\right)$, we use the Euler-Lagrange equation on (24) and we get

$$\begin{aligned} P_1'(\phi) &= \frac{\partial f}{\partial \phi}(x, \phi, \nabla\phi) - \frac{d}{dx}\left(\frac{\partial f}{\partial \phi_x}(x, \phi, \nabla\phi)\right) - \frac{d}{dy}\left(\frac{\partial f}{\partial \phi_y}(x, \phi, \nabla\phi)\right) \\ &= -\phi_{xx}^2 - \phi_{yy}^2 \\ &= -\triangle\phi, \end{aligned} \tag{25}$$

where $\triangle$ is the Laplace operator. For $P_2(\phi)$ we get

$$\begin{aligned} P_2'(\phi) &= -\frac{d}{dx}\frac{\phi_x}{\sqrt{\phi_x^2 + \phi_y^2}} - \frac{d}{dy}\frac{\phi_y}{\sqrt{\phi_x^2 + \phi_y^2}} \\ &= -\text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right). \end{aligned} \tag{26}$$

The last integral in (24) vanishes using the Euler-Lagrange equation. Therefore the derivative of the penalizing term is

$$P'(\phi) \;=\; -\triangle\phi + \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{27}$$

To compute $L'(\phi)$, see (20), we need

$$\frac{\partial}{\partial\phi}\left(g(x)\delta(\phi)|\nabla\phi|\right) \;=\; g(x)\delta'(\phi)|\nabla\phi| \tag{28}$$

and

$$-\sum_{x_i\in\{x,y\}}\frac{d}{dx_i}\left(\frac{\partial}{\partial\phi_{x_i}}\left(g(x)\delta(\phi)|\nabla\phi|\right)\right) = -\operatorname{div}\left(g(x)\delta(\phi)\frac{\nabla\phi}{|\nabla\phi|}\right), \tag{29}$$

where we used (26). Let u be a scalar function, v a vector function. We will use the following identity for the divergence of their product:

$$\operatorname{div}(uv) = \nabla u\cdot v + u\operatorname{div} v, \tag{30}$$

where $\nabla u\cdot v$ is their inner product. Denoting $u=\delta(\phi)$ and $v=g(x)\frac{\nabla\phi}{|\nabla\phi|}$, we use (30) and obtain

$$\operatorname{div}\left(g(x)\delta(\phi)\frac{\nabla\phi}{|\nabla\phi|}\right) \;=\; \nabla\left(\delta(\phi)\right)\cdot\left(g(x)\frac{\nabla\phi}{|\nabla\phi|}\right) + \delta(\phi)\operatorname{div}\left(g(x)\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{31}$$

We will further focus on the first term on the right hand side of the equation. Because $g(x)$ is scalar, we can rewrite the term as follows:

$$\begin{aligned}\nabla\left(\delta(\phi)\right)\cdot\left(g(x)\frac{\nabla\phi}{|\nabla\phi|}\right) &= g(x)\delta'(\phi)\frac{\nabla\phi\cdot\nabla\phi}{|\nabla\phi|}\\ &= g(x)\delta'(\phi)|\nabla\phi|.\end{aligned} \tag{32}$$

Substituting (32) into the divergence term (31), then into (29) and together with (28) into the Euler-Lagrange equation (7) we obtain

$$L'(\phi) = -\delta(\phi)\operatorname{div}\left(g(x)\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{33}$$

The terms (28) and (32) are subtracted.

The last term in our functional is the area term $A(\phi)$, see (21). Using the Euler-

Lagrange equation (7) we obtain

$$A'(\phi) \;\; = \;\; -g(x)\delta(\phi). \tag{34}$$

Substituting the derivatives of the individual functional terms into the functional we get the derivative of the total energy, its negative being the desired gradient flow (23).

The *gradient flow* is then defined by

$$\begin{aligned} \frac{\partial \phi}{\partial t} = -E'(\phi) \;\; &= \;\; \mu \left[\triangle \phi - \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right] + \\ &+ \lambda \delta(\phi) \operatorname{div} \left(g(x) \frac{\nabla \phi}{|\nabla \phi|} \right) + \alpha g(x) \delta(\phi). \end{aligned} \tag{35}$$

To improve the speed and accuracy of the evolution we have incorporated new terms to the gradient flow suggested in [8].

The first term is called the *data term* $D(\phi)$. The data term uses the mean values of image intensity of areas currently assigned as background, denoted as m_1, or objects, denoted as m_2. Consequently, it drives the motion of the level set curve of ϕ towards one of them, depending on whether the intensity of a point is more similar to either m_1 or m_2. The data term is defined by

$$D(\phi) \;\; = \beta \delta(\phi) \left(- \left(I - m_1 \right)^2 + \left(I - m_2 \right)^2 \right), \tag{36}$$

where $\beta > 0$ is a weighting parameter, δ is the delta distribution (18). The delta distribution selects only the zero level curve from the domain of ϕ. Since we assume that values of ϕ are negative inside the zero level curve and positive outside (8), we either add or subtract the difference between the intensity and the mean intensities (36), which speeds up the evolution.

Since our application is the segmentation of cell images, we can assume similar features of the images. The typical microscopic cell image we encounter is shown in the Figure 4. The cell center is of a similar colour as the background and forms a visual edge inside the cell. As a consequence, a contour may form inside the cell, which is shown in the Figure 5. To avoid the undesirable interior contour we introduce a new term, which we call the *variance term V*. It works on the assumption

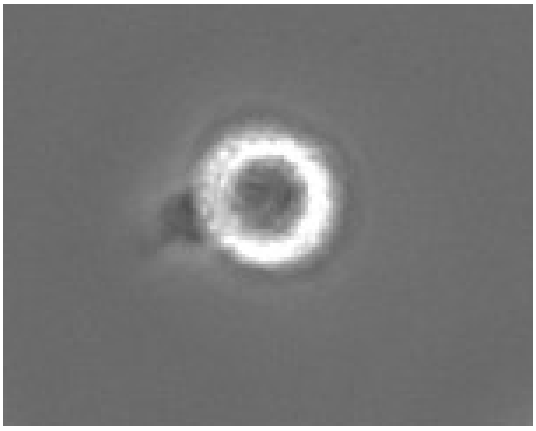


Figure 4: Typical image of a cell.

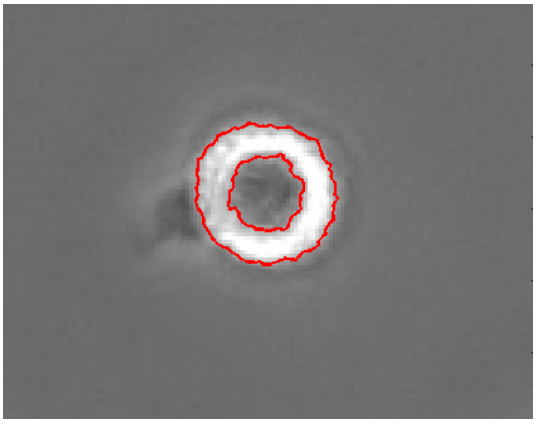


Figure 5: An undesirable contour inside a cell.

that the variance of background is significantly smaller than variance of the cells. It is defined similar to the data term but instead of the mean values of image intensity it uses the mean values of variance and compares it with the variance of a given point. The variance term is then

$$V(\phi) = \rho\delta(\phi)\left(-\left(Var - v_1\right)^2 + \left(Var - v_2\right)^2\right), \tag{37}$$

where $\rho > 0$ is a weighting parameter, Var is the variance of the image on a neighbourhood of a given point, v_1 is the mean exterior variance and v_2 is the mean interior variance.

Definition. The gradient flow is then defined by

$$\begin{aligned}\frac{\delta\phi}{\delta t} = -E'(\phi) \;\; = \;\; & \mu\left[\triangle\phi - \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right] + \lambda\delta(\phi)\operatorname{div}\left(g(x)\frac{\nabla\phi}{|\nabla\phi|}\right) + \\ & + \alpha g(x)\delta(\phi) + \beta\delta(\phi)\left(-\left(I - m_1\right)^2 + \left(I - m_2\right)^2\right) + \\ & + \rho\delta(\phi)\left(-\left(Var - v_1\right)^2 + \left(Var - v_2\right)^2\right). \end{aligned} \tag{38}$$

This is the equation we will solve by finite differences in the next chapter. The equation with suitable parameters also leads to the results presented in the last chapter.

3 Implementation

The first section of this chapter describes the numerical scheme used for calculations. In the second section we present three different approaches of initialization and we focus on the optimization of parameters in the last section.

3.1 Numerical scheme

In practice, the delta distribution (18) and the Heaviside function (17) are approximated by the smooth functions defined by

$$\delta_\epsilon(x) = \begin{cases} 0, & |x| > \epsilon, \\ \frac{1}{2\epsilon}\left[1 + cos\left(\frac{\pi x}{\epsilon}\right)\right], & |x| \leq \epsilon, \end{cases}$$

$$H_\epsilon = \begin{cases} 0, & x < -\epsilon, \\ 1, & x > \epsilon, \\ \frac{1}{2}\left(1 + \frac{x}{\epsilon} + \frac{1}{\pi} sin\left(\frac{\pi x}{\epsilon}\right)\right), & |x| \leq \epsilon. \end{cases}$$

For all our experiments we set the parameter ϵ to 1, which is the width of a pixel.

The derivatives of ϕ are approximated by finite differences. We use the central and forward differences, which are defined in one dimension as follows, see [6].

Definition. Let $f : \mathbb{R} \to \mathbb{R}$ be a function and $h > 0$.

The *forward difference* is given by

$$\triangle_h[f](x) = f(x + h) - f(x).$$

The derivative of f approximated by the forward difference is then

$$\frac{\partial f(x)}{\partial x} \cong \frac{\triangle_h[f](x)}{h} = \frac{f(x + h) - f(x)}{h} \tag{39}$$

The *central difference* is defined as

$$\delta_h[f](x) = f(x + \frac{1}{2}h) - f(x - \frac{1}{2}h),$$

the derivative approximated by the central difference is defined analogically as (39).

The spatial derivatives $\frac{\partial \phi}{\partial x}$ and $\frac{\partial \phi}{\partial y}$ are approximated by the central difference with fixed space steps $h = \triangle x = \triangle y = 1$. The time derivative $\frac{\partial \phi}{\partial t}$ is approximated with the forward difference.

The level set function $\phi(t, x, y)$ is discretized as $\phi^k_{i,j}$, where (i, j) is a space index and k is a time index. Then, the level set evolution using the gradient flow (38) can be written as

$$\frac{\phi^{k+1}_{i,j} - \phi^k_{i,j}}{\tau} = F(\phi^k_{i,j}), \tag{40}$$

where $F(\phi^k_{i,j})$ is the approximation of the right hand side of (38) and τ is a time step. The difference equation (40) can be expressed as an *iteration process*.

Definition. The iteration process used in the numerical implementation is

$$\phi^{k+1}_{i,j} = \phi^k_{i,j} + \tau F(\phi^k_{i,j}).$$

Computation of the Gradient and Laplace operator

We speed up the evaluation of the gradient and Laplace operator using new functions, based on the central differences, written in the C language. For computing the gradient, we used the code available online, written by Jan Simon [18]. We modified this code to implement the Laplace operator. The new implementations are about three times faster than the standard functions in MATLAB.

3.2 Initialization of the Level Set Function

In the traditional level set formulation it is important to initialize the level set function ϕ as a signed distance function ϕ_0. If the initial function significantly differs from a signed distance function, the re-initialization processes are not able to re-initialize the function to a signed distance function [8].

In our formulation the re-initialization process can be omitted due to the new penalizing term (16). We also use initial functions that are significantly different from a signed distance function. The penalizing term may not be able to maintain the level set function ϕ as an approximation of a signed distance function globally on the entire image domain but it keeps it as the approximation locally near the zero level curve.

In our first numerical experiments, we used an initial function similar to the one proposed in [8]. Since it was too slow, we suggested two different approaches.

The best results come from the initialization by thresholding the image of variance described later in this section. This approach is remarkably fast.

3.2.1 The Original Initialization

Let Ω_0 be a subset in the image domain Ω. We define the initial level set function ϕ_0 by

$$\phi_0 = \begin{cases} -1, & (x, y) \in \Omega_0, \\ 1, & (x, y) \in \Omega - \Omega_0. \end{cases}$$

Since we assume initialization outside the cells and set the weighting parameters according to that, see Section 2.1.4, we define the region Ω_0 as a rectangle 8 pixels smaller than the image, which is centered in the image.

As an example of this initialization, we present numerical results of segmentation in Figure 6. Although the image is only a cutout, it needs too many iterations to reach the result. The parameters in the gradient flow (38) were chosen based on numerical experiments to achieve the best possible accuracy and speed of the evolution. We cut the image of a cluster of cells from an original image. The size of the original image is 960×1280 pixels, the cutout has 216×206 pixels. With the initialization described in this section, the algorithm needs 1650 iterations to reach the boundary of the cluster. Therefore we suggested more efficient initializations, which are described in the following sections.

3.2.2 Initialization by the Previous Images

The images we process capture the evolution of cells. We have a sequence of images, each is taken two minutes after the previous one. Therefore we can assume that the

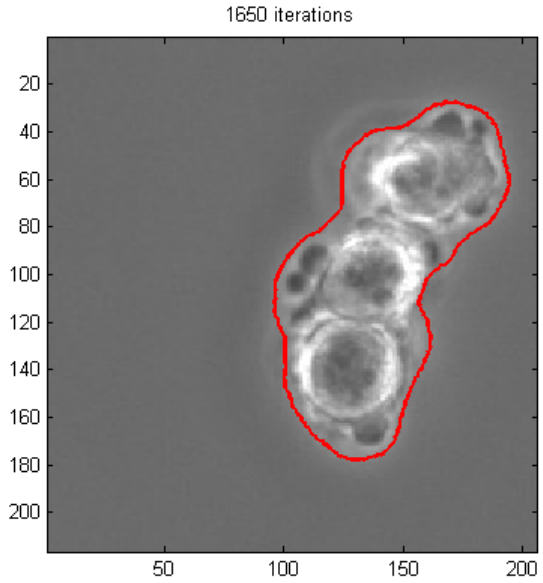


Figure 6: Result of the original initialization using manually optimized parameters.

consecutive images do not differ significantly. As an initialization of an image, we can take the resulting level set function of the segmentation of the previous image. The level set function can be used unmodified, or we can keep the zero level curve and set the region Ω_0 as the interior of the cells and use the previous method of initialization, i.e. set the interior as -1 and the exterior as 1. In the experiments we made, there is no significant difference between the results using modified or unmodified level set function.

For the following numerical experiments, we used the result in Figure 6 for the initialization. Each of the next four images was initialized with the same initial function, derived from the first image, to better understand the influence of the movement of the cell cluster. The initial level set function was set as a piecewise constant function described in the previous paragraph.

The parameters in the experiment of Figure 7 are the same as we used for computing Figure 6. Although the iteration process converges after about 200 iterations, which is about eight times faster than with the previous initialization, the results are not precise. The problem is that a part of the initial curve lays inside the cell cluster. We assume that the initial curve would be either inside or outside of the cell, or generally of the object we seek. According to the position of the initial zero curve, we set the sign of the parameter of the area term (21). Since the initial zero curve lays partially inside the cluster, during the evolution the process captures the

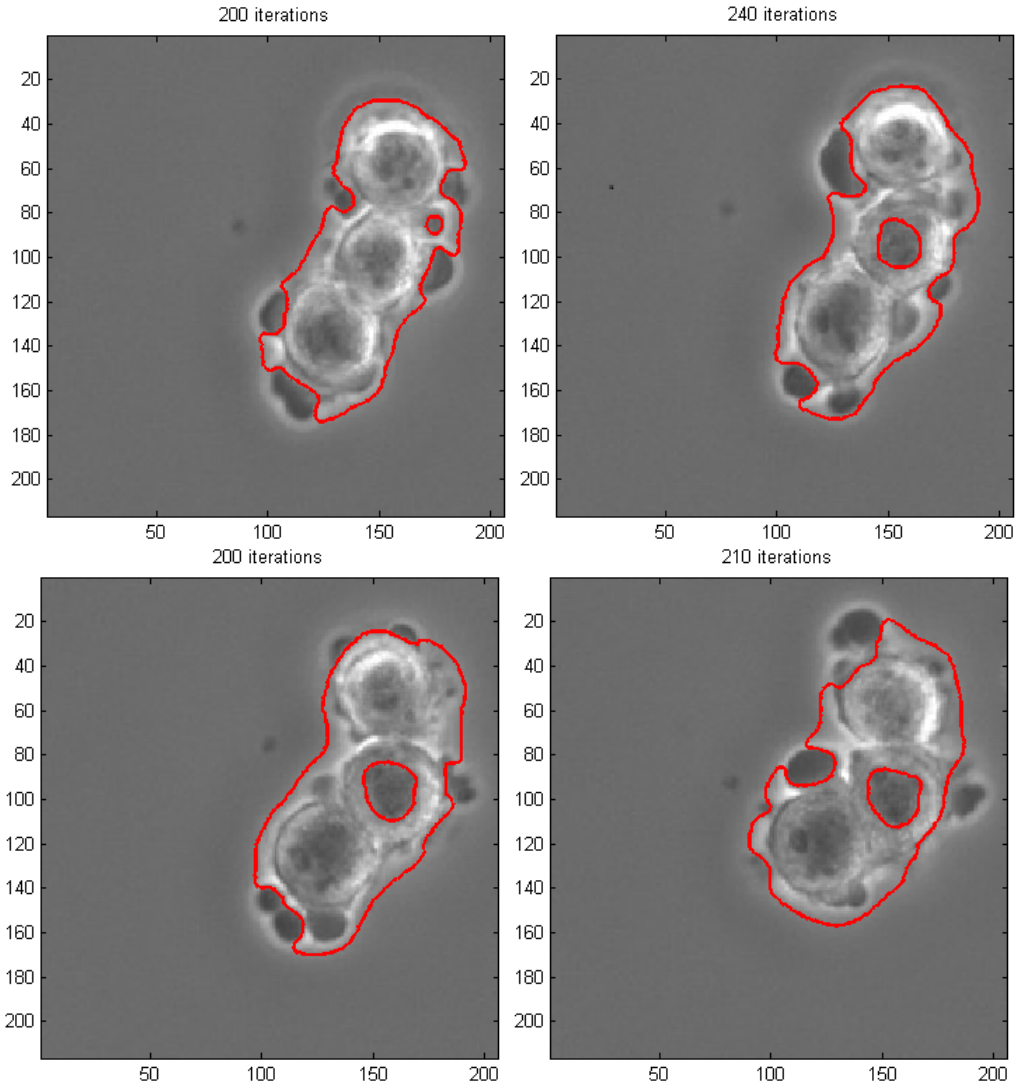


Figure 7: Initialization by the result of the first image using the same parameters as were used in Figure 6.

interior edges and mistakenly labels the cell interior as a background.

To avoid the interior curves, we set new parameters, especially magnifying the variance term (37). The results are shown in Figure 8. The curve converges after about 200 iterations but we let the iteration process compute at least 1000 iterations to assure that the interior curves do not appear later in the process. Although there are no interior curves, some parts of the cells are not included in the result and therefore the result is not accurate enough.

If we use the unchanged zero level curve of the result as an initialization for the consecutive image, where the cells moved, it is typical that part of the zero curve would be inside the cell region and part outside of it. We suggest *dilation* as an

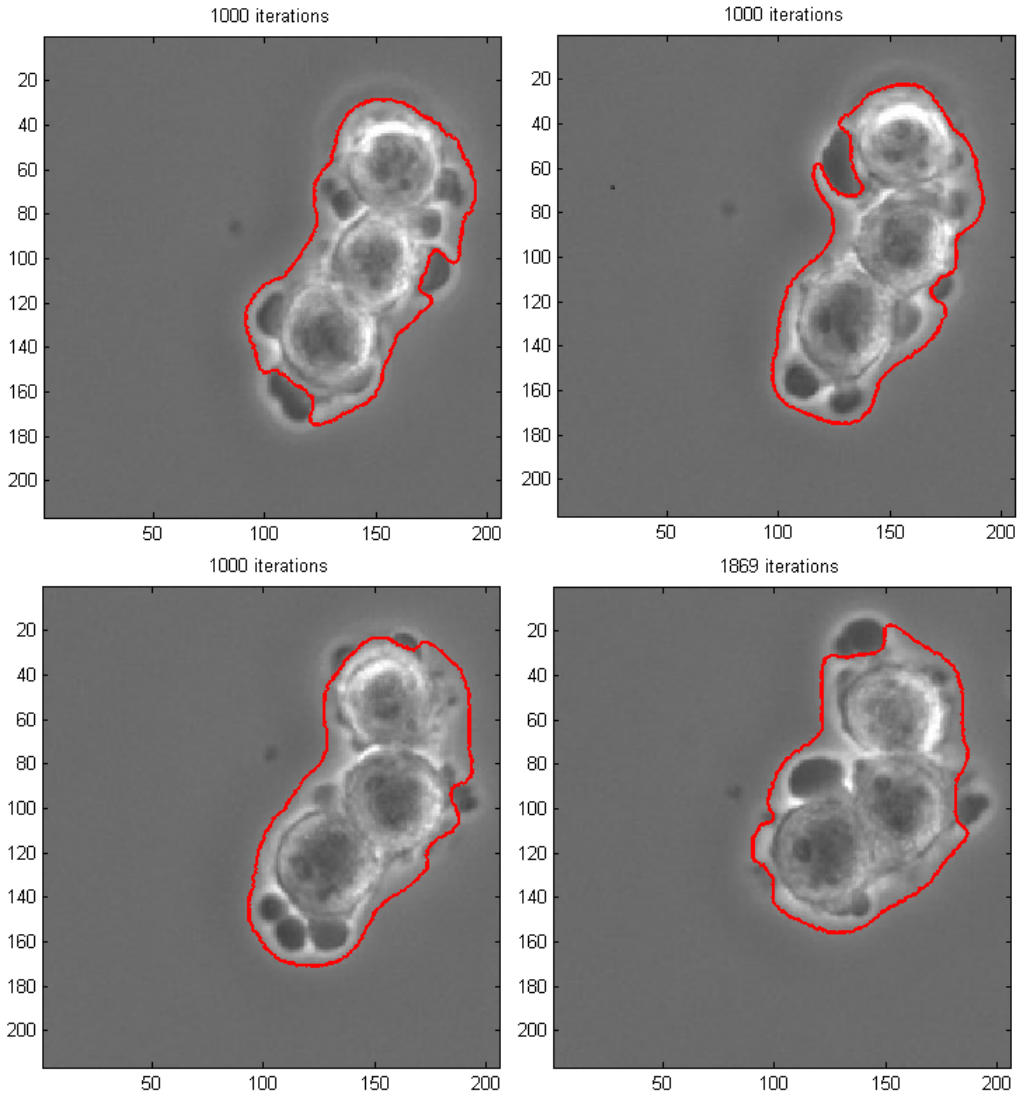


Figure 8: Initialization by the result of the first image using new parameters magnifying the variance term.

option of a numerical remedy. Dilation is a process that enlarges an object [13]. Thus the initial curve would be outside of the cell and we would avoid the encountered problems. However, incorporating dilation into the algorithm would cause other problems with implementation. Therefore we present a simpler approach described in the next section.

3.2.3 Initialization by Variance

The region based initialization proposed by [8] and described in Section 3.2.1 is flexible for various applications. If we had an approximation of the regions of interest, we could use this approximation to construct the initial level set function ϕ_0.

The method to roughly separate the objects from the background could be *thresholding*. Thresholding is the simplest segmentation method.

Definition. *Thresholding* is a function $T : \mathbb{R}^2 \to \mathbb{R}$ defined as

$$T(c_{(i,j)}) = \begin{cases} 0, & \text{for } c_{(i,j)} < p, \\ 255, & \text{for } c_{(i,j)} \geq p, \end{cases} \tag{41}$$

where $c_{(i,j)}$ is the value in the pixel (i, j) and p is a given threshold.

The value $c_{(i,j)}$ is usually the intensity or colour because we can assume that the objects are of a different colour than the background. Thresholding thus replaces each pixel with either a black or white pixel depending on whether the intensity is greater or smaller than the threshold p.

We assume the image background to have constant intensity with some insignificant perturbations caused by noise. However, the cell centers have similar intensity as the background. Therefore, thresholding of the original image would not be useful as it would label the cell centers as the background.

The method we suggest utilizes the same property as when we introduced the variance term (37), i.e. the assumption that the background has significantly smaller variance than the cells.

For each pixel we compute the variance of its neighborhood and then we use thresholding on the image of variance. As a result we obtain a black and white image which is then used as the initialization, the level set function ϕ in the cell regions is set to -1, in the background it is set to 1.

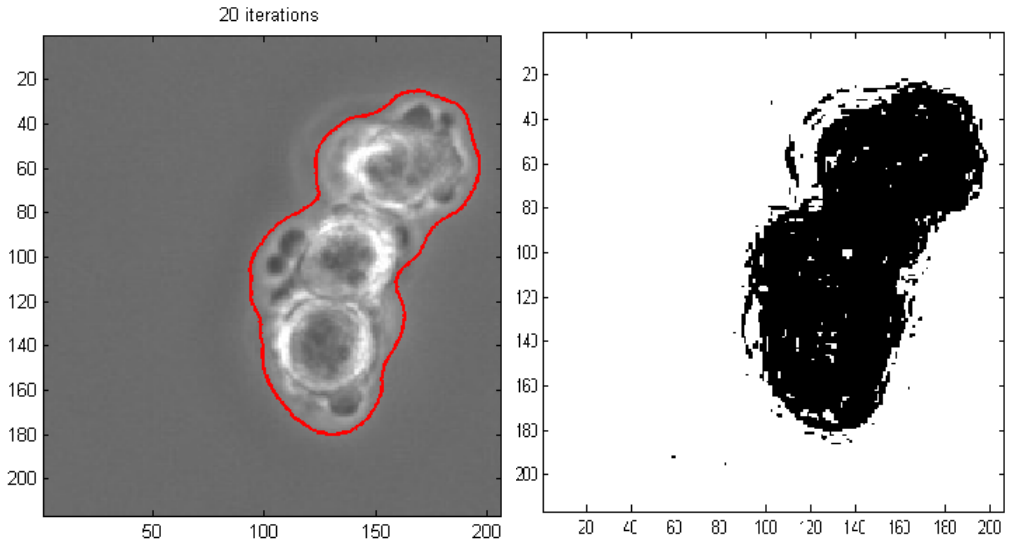


Figure 9: Initialization by thresholding the image of variance. Threshold $p = 2.2$ was found manually.

As an example, we use the same image and the same parameters as in Section 3.2.1 with an initialization given by thresholding. The result is shown in Figure 9. We obtain a similar result as in Figure 6 but it only takes 20 iterations, which is more than 80 times less than with the original initialization.

The second image in Figure 9 shows the binary image obtained by thresholding with the threshold $p = 2.2$. Although the initial level set function ϕ_0 has many zero level curves for only one cell cluster, due to the formulation of the gradient flow the small curves disappear after about 15 iterations.

3.3 Automatic Optimization of the Weighting Parameters

The sequences of images of cells we process may contain thousands of images. The cells may grow, proliferate or die and the images in the beginning of a sequence may have a completely different structure than the images by the end of the sequence. Figure 10 shows the first and the last image of the sequence we will process. In the first image, there are isolated cells but in the last image the background is almost completely covered by cells. Therefore, it is challenging to find universal parameters for all images in the sequence.

The parameters can be selected manually, based on an observation of numerical experiments to achieve the best possible accuracy and speed of the evolution. We

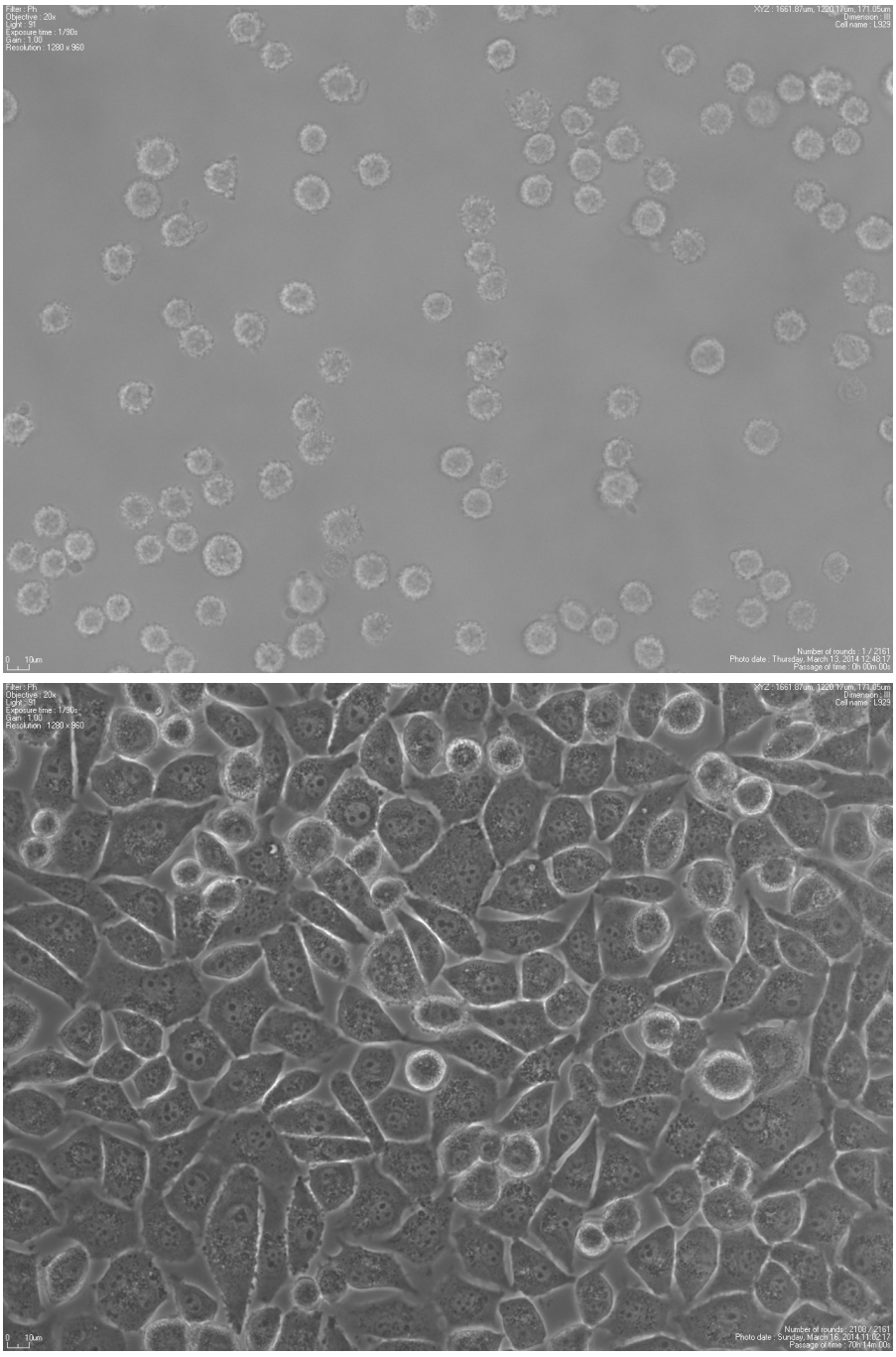


Figure 10: The difference between the first and the last image of a sequence of 2161 images. There are isolated cells on a background in the first image while the majority of the last image is covered with cells.

used manual optimization of the parameters in our initial experiments. However, this method of optimizing can be a tedious work. Therefore, we used several *automatic methods of optimization* in MATLAB to see whether the automatic methods are applicable for our problem.

The most accurate results were computed by the trust-region-reflective method, see [3]. The method minimizes the difference between the results of the segmentation and the ground truth, i.e. images segmented manually by specialists.

In the next sections we present numerical experiments on selected images from a sequence of 2161 images. We applied the automatic optimization only on the first image of the sequence and then we used the resulting parameters for all of the remaining images. The values of the parameters we used in our experiments are

$$\mu = 0.2,\ \lambda = 479.7927,\ \alpha = 0.1114,\ \beta = 0.0062,\ \rho = 0.0175,\ p = 2.0439, \qquad (42)$$

where the first five parameters are the weighting parameters in the gradient flow (38) and the last parameter is the threshold, see (41).

4 Numerical Experiments

This chapter adresses analysis of numerical experiments on the real data. The numerical experiments and visualizations of the results were performed using MATLAB. We used the parameters described in (42).

Our images come from The laboratory of tissue culture, Institute of complex systems, Faculty of Fisheries and Protection of Waters, University of South Bohemia in České Budějovice. We chose 50 images from a sequence of over 2000 images. The results from our method are compared with manually segmented images, which are taken as the ground truth. We use the F_1 *score* for evaluation. Our results are also compared with the initialization by variance and with the results of Soukup's algorithm described in [19].

However, the manual evaluation depends on the subjective perspective. For the experiments in [19], a sequence of images was manually processed by two different laboratory technicians. The similarity of the manual segmentation was only 94.8%, which means, that 94.8% of pixels was assigned to the same category, i.e. either as the area of cells or background. Our images come from the same laboratory as in [19], so we can assume similar accuracy.

4.1 Number of iterations

As in every iteration process, we need to specify a stopping criteria or a number of iterations. If the exact solution was found, the evolution would stop which means that the terms in gradient flow (38) would be zero.

We will analyze the change of each term with the number of iterations. We show the values of some of the terms from the gradient flow in Figure 11. The horizontal axis represents the number of iterations, the vertical axis represents the value of the

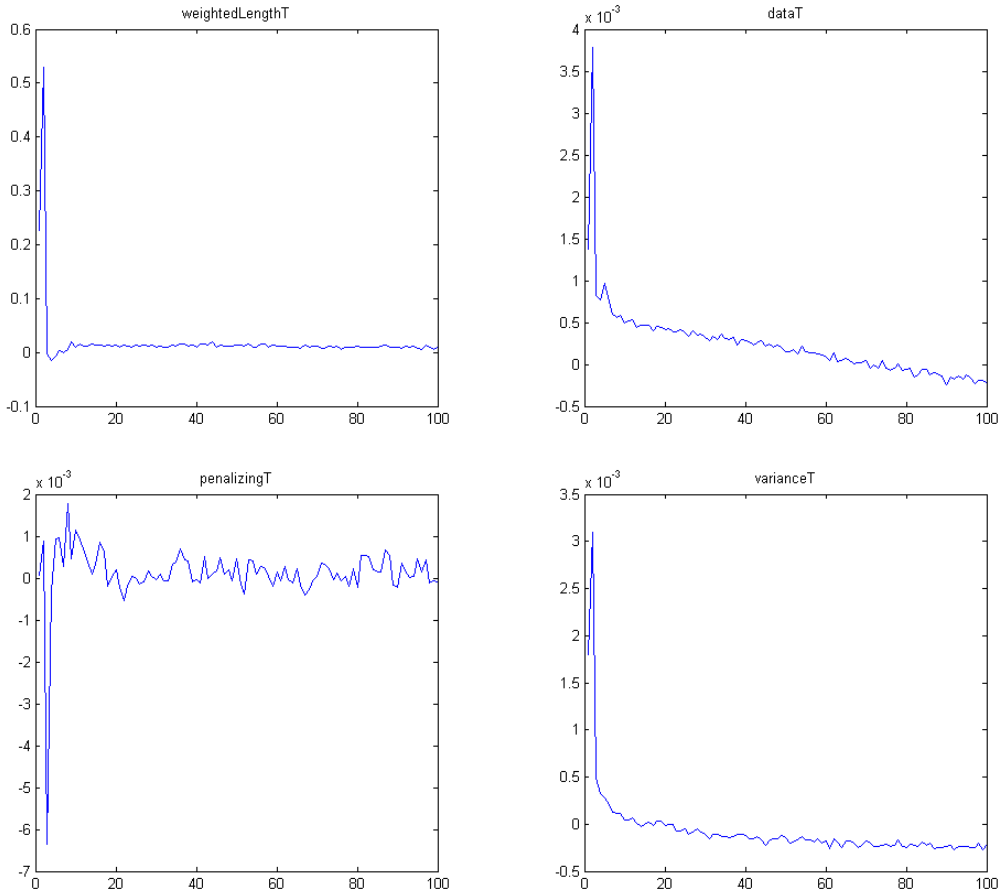


Figure 11: The value of the terms in the gradient flow. The horizontal axis represents the number of iterations, the vertical axis represents terms in the gradient flow.

term in the gradient flow. The image we process in this Figure is the first image from the sequence, it is shown in Figure 10.

We can see that the terms are almost a constant zero after about 10 or 15 iterations, so the minimum number of iterations for the method to converge appears to be 15 iterations.

In the experiments, we used a various number of iterations - from 10 up to 5000 iterations. In the next section we will present the results after 15-120 iterations.

4.2 Definition of the F_1 score

We will use the F_1 *score* for evaluation of the results. The F_1 *score* considers *precision* and *recall* of the test to compute its relevance. In the image segmentation context, precision and recall are defined in the terms of a set of retrieved pixels and a set of relevant pixels, which are in our case the pixels of cells. Precision is the fraction of

retrieved pixels that are relevant and recall is the fraction of relevant pixels that are retrieved.

Let us define a *confusion matrix* in Table 1. Confusion matrix divides the image pixels into four sets depending on the outcome of the algorithm comparing to the ground truth. We will use the terms in confusion matrix in the following definition of precision, recall and F_1 *score*.

		Ground truth	
		Cell	Background
Algorithm outcome	Cell	*True positive (TP)*	*False positive (FP)*
	Background	*False negative (FN)*	*True negative (TN)*

Table 1: Confusion matrix

Definition. *Precision p:*

$$p = \frac{TP}{TP + FP}.$$

Recall r:

$$r = \frac{TP}{TP + FN}.$$

F_1 score:

$$F_1 = 2 \cdot \frac{p \cdot r}{p + r}. \tag{43}$$

The *F_1 score* is the harmonic mean of precision and recall. The best results take the value 1 while the worst results has the score 0.

4.3 Results

The sequence of images we are processing has 2161 images. Since we need manually segmented images from specialists as the ground truth and since it is time consuming to get them, we chose 50 representative images from this sequence to analyze. The first image of the sequence was used for optimization of parameters. The remaining images were segmented by our algorithm and compared with the initialization by variance and also with results arising from the algorithm by Soukup, described in [19]. Threshold in the initialization by variance was chosen as the result of the automatic optimization (42). For the Soukup's algorithm we used its standard parameters. The results were compared using the *F_1 score* (43).

F_1 of our algorithm		F_1 of Soukup's algorithm	F_1 of initialization
15 iterations	0.8717	0.8655	0.8390
30 iterations	0.8767		
80 iterations	0.8835		
100 iterations	0.8842		
120 iterations	0.8842		

Table 2: Comparison of the F_1 *scores*

One problem of our images is the text at the corners of the images. The text specifies each microscopical image and the method finds the edges and evaluates the text as a cell. We therefore evaluate the F_1 *score* on cut images without the text at the corners. The area we cut out is 2% of the image.

As described above, we used manually segmented images as the ground truth. The ground truth was used for computing the F_1 *score* for each of the three groups of results: our algorithm, Soukup's algorithm and the initialization by variance. We then computed the mean value of the F_1 *scores* of the 49 images for each algorithm which is shown in Table 2. We used different number of iterations for our algorithm which is also presented in Table 2.

Table 2 shows that our initialization by variance is already very efficient as its F_1 *score* is 0.8390. Therefore with only 15 iterations we get the F_1 *score* of 0.8717 which is higher than the F_1 *score* of 0.8655 of Soukup's algorithm. From Table 2 we can see that the F_1 *score* of our algorithm grows with the number of iterations until it reaches 100 iterations. The F_1 *score* of our algorithm after 100 iterations is 0.8842 which is the same score as using 120 iterations. We can assume the convergence of the algorithm after 100 iterations. The segmentation using 100 iterations lasts less than 100 sec.

We show some of the resulting images segmented by our algorithm in Figures 12-16. The first three figures show images from the beginning of the sequence we process. The cells on these images are still isolated and the background covers majority of the image. The red curve is the zero level of the level set function. After 15 iterations some interiors of the cells are assigned as the background, therefore we show the results using 100 iterations, where this problem is solved due to the evolution and the variance term. After 100 iterations the curve is also smoother because of the length and area terms. The last two figures show images from the end of the sequence. These images are covered mostly by cells. The interior of the closed curves in Figures 15 and 16 thus represent the background and the exterior is the area of cells.

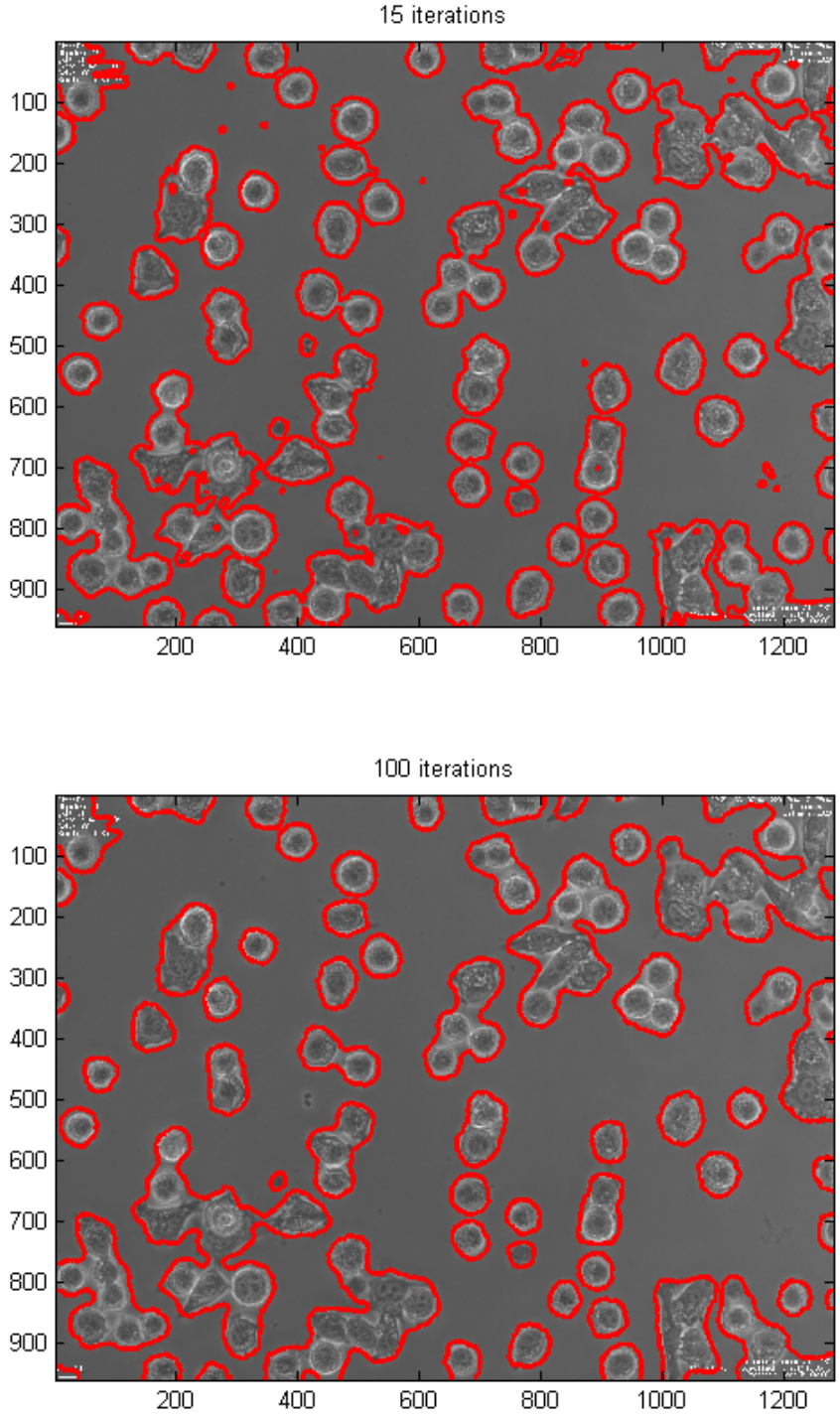


Figure 12: The segmentation of the 70^{th} image of the sequence. The first image shows the result using 15 iterations, the second one is the result using 100 iterations.

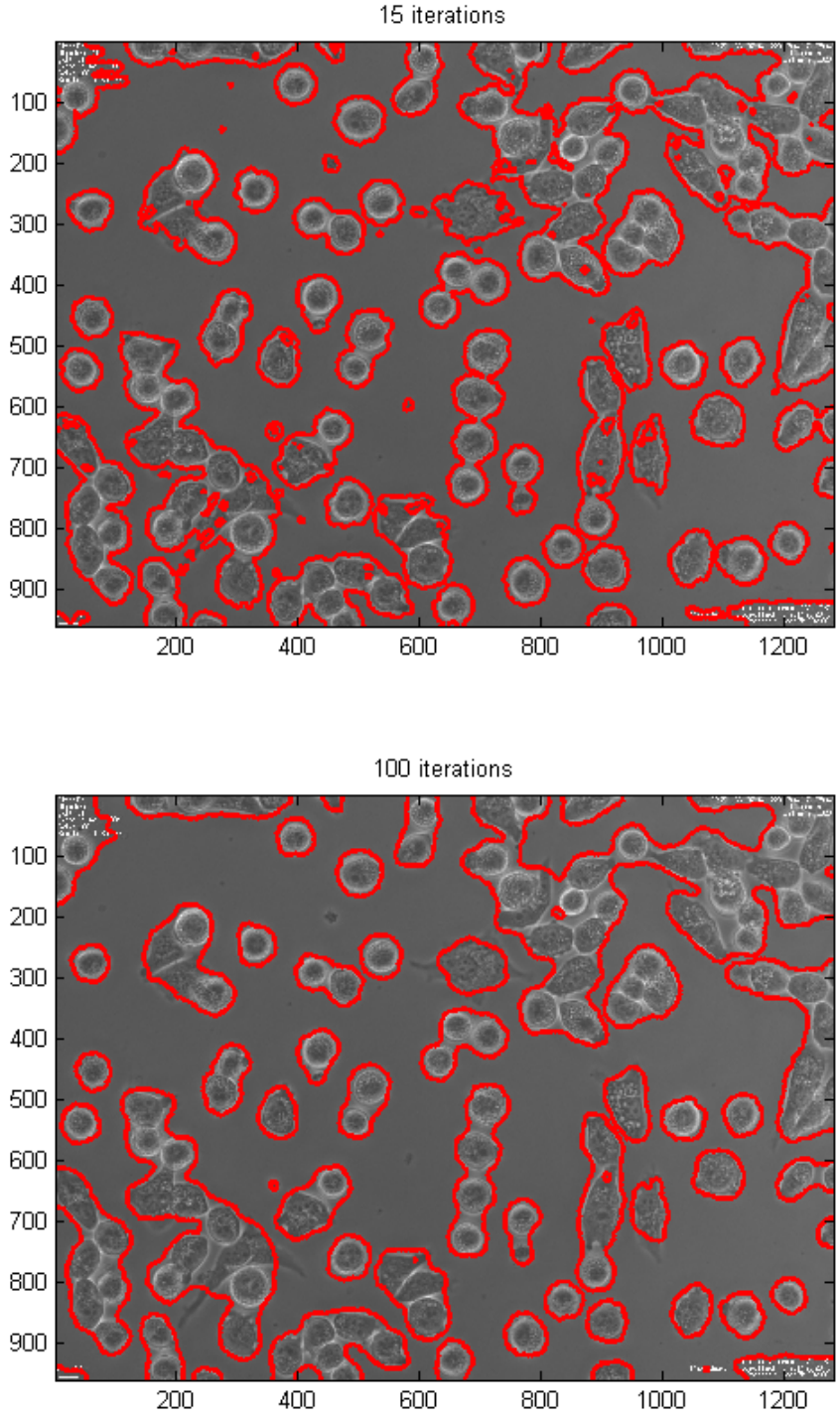


Figure 13: The segmentation of the 227$^{\text{th}}$ image of the sequence. The first image shows the result using 15 iterations, the second one is the result using 100 iterations.

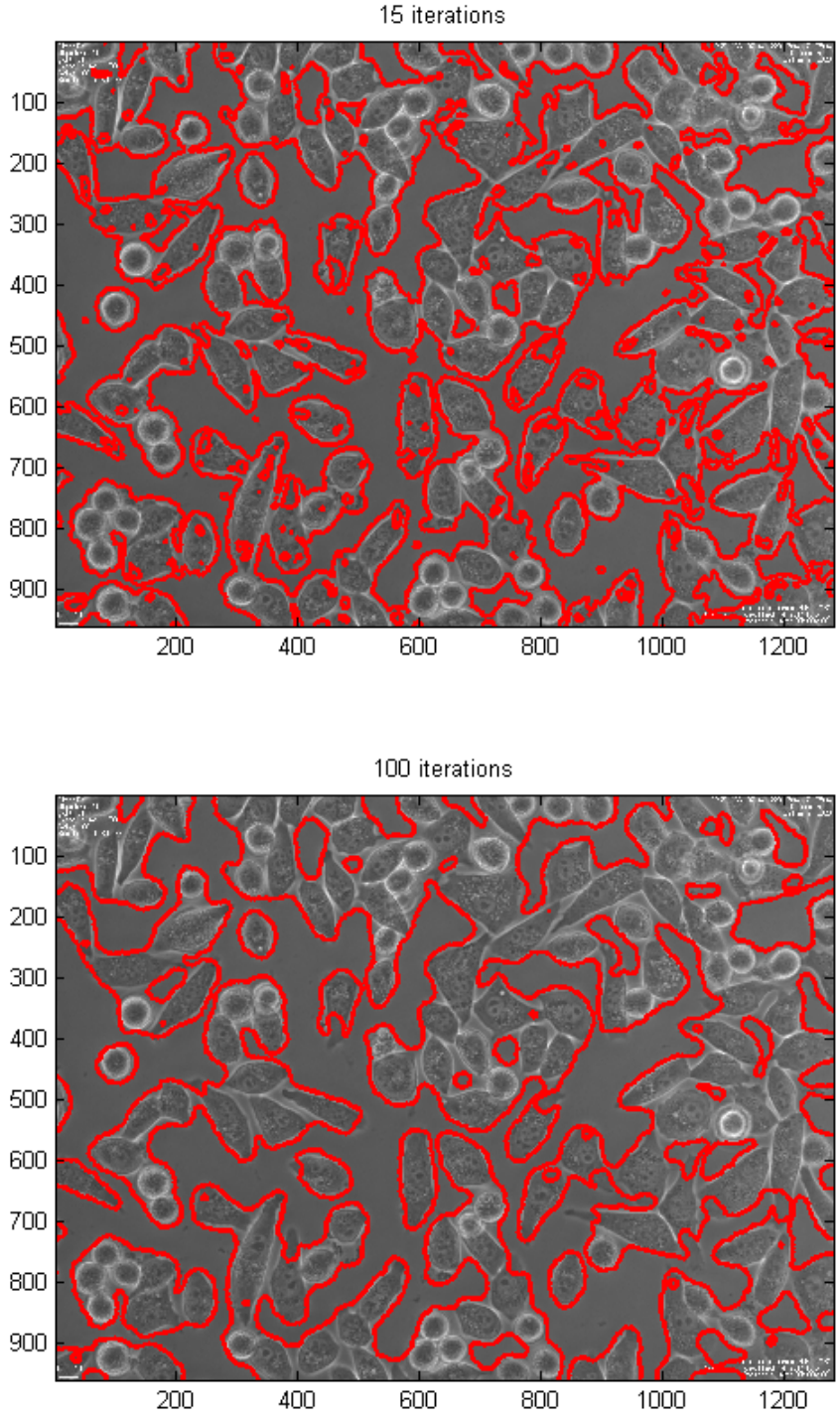


Figure 14: The segmentation of the 904^{th} image of the sequence. The first image shows the result using 15 iterations, the second one is the result using 100 iterations.

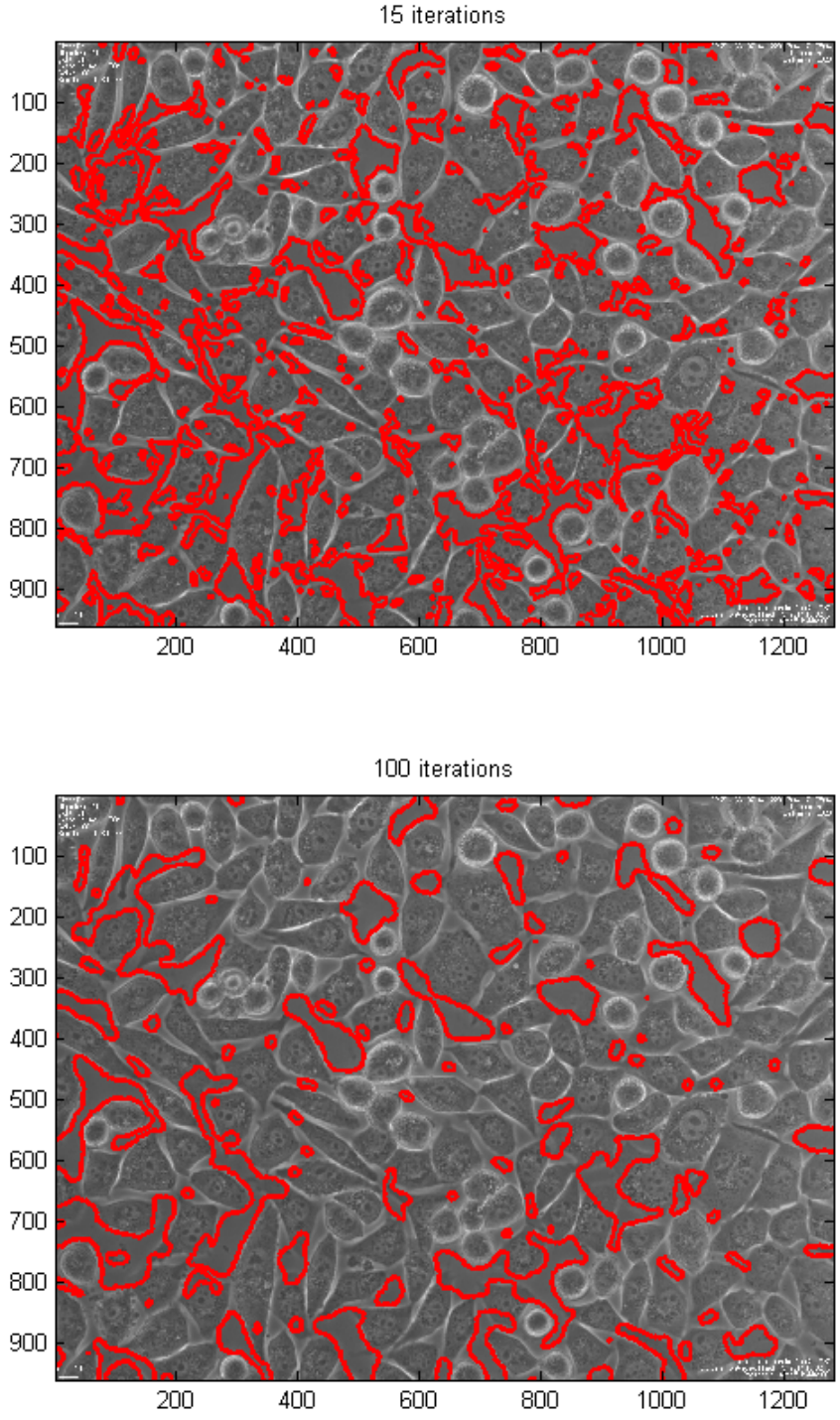


Figure 15: The segmentation of the 1553rd image of the sequence. The first image shows the result using 15 iterations, the second one is the result using 100 iterations.

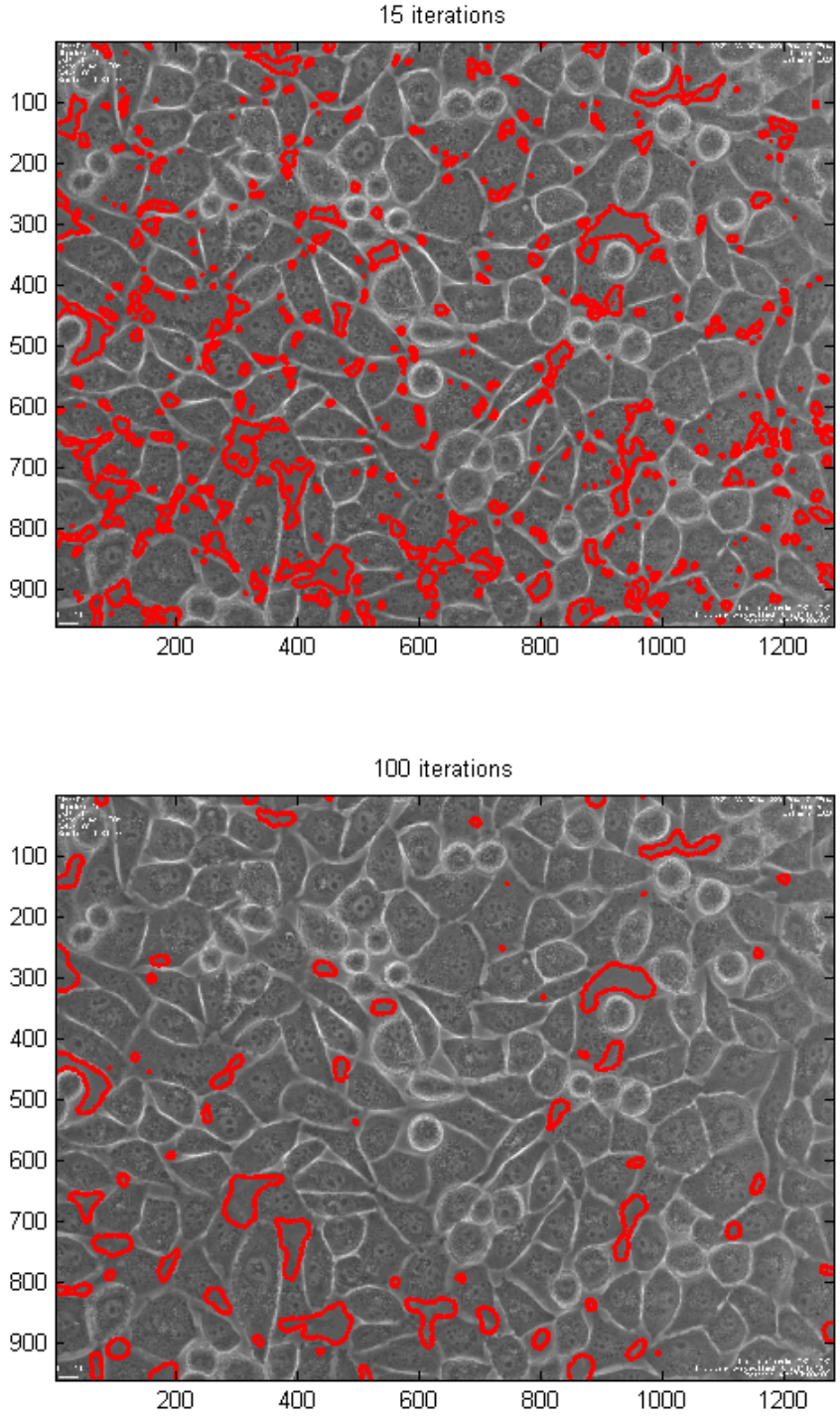


Figure 16: The segmentation of the last image of the sequence. The first image shows the result using 15 iterations, the second one is the result using 120 iterations.

5 Conclusion

In this thesis, we studied the level set methods and its application in the research of new materials for implants. We presented a mathematical background for the level set methods and we introduced a new method of segmentation. We processed microscopic images of cancer cells and the results were compared with the manual segmentation.

The algorithm we proposed is based on the variational formulation of the level set methods, which is a problem of minimizing a functional. The method is based on the functional presented in [8], which describes a level set function. The level set function implicitly describes an interface dividing an image into the background and cells. The functional is minimized by a gradient flow. We introduced new terms in the gradient flow to achieve higher accuracy and faster convergence.

We introduced a region-based initialization. The initialization is computed by thresholding the image of local variance. This initialization significantly reduces the necessary number of iterations for the method to converge.

We used a finite difference scheme for discretization. Instead of using standard MATLAB functions for evaluation of the gradient and Laplace operator, we used functions written in the C programming language compiled to MATLAB, which are about three times faster.

The parameters were optimized using automatic optimizing algorithms in MATLAB.

We compared our results with manually processed images and compared its F_1 *scores* with another algorithms. The results of our algorithm are better than the compared results.

References

[1] T. Chan, J. Shen, "Image Processing And Analysis: Variational, Pde, Wavelet, And Stochastic Methods", *SIAM*, 2005.

[2] T. Chan, L. Vese, "Active Contours without edges", *IEEE Trans. Imag. Proc.*, vol. 10, pp. 266-277, 2001.

[3] A.R. Conn, N.I.M. Gould, P.L. Toint, "Trust-Region Methods", *Society for Industrial and Applied Mathematics*, 2000.

[4] R.C. Gonzalez, R.E. Woods, "Digital Image Processing", *Prentice Hall*, 3rd edition, 2007.

[5] G. Grubb, "Distributions and Operators", *Springer*, 2009.

[6] H. Levy, F. Lessman, "Finite Difference Equations", *Dover*, 1992.

[7] F. Li, X. Zhou, H. Zhao, S. T.C. Wong, "Cell Segmentation Using Front Vector Flow Guided Active Contours", *Med Image Comput Comput Assist Interv.*, pp. 609-616, 2009.

[8] Ch. Li, Ch. Xu, Ch. Gui, M.D. Fox, "Level set evolution without re-initialization: a new variational formulation", *IEEE Computer Society Conference*, vol. 1, pp. 430-436, 2005.

[9] K. Li, E.D. Miller, L.E. Weiss, P.G. Campbell, T. Kanade, "Online Tracking of Migrating and Proliferating Cells Imaged with Phase-Contrast Microscopy", *IEEE Conference on Computer Vision and Pattern Recognition Workshop,* 2006.

[10] B.M. Mehtre, B. Chatterjee, "Segmentation of fingerprint images—a composite method", *Pattern recognition,* pp. 381-385, 1989.

[11] M. Nixon, A. Aguado, "Feature Extraction & Image Processing", *Academic Press*, 2nd edition 2008.

[12] S. Osher, R. Fedkiw, "Level Set Methods and Dynamic Implicit Surfaces (Applied Mathematical Sciences)", *Springer,* 2003.

[13] S. Osher, J.A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations", *J. Comp. Phys.*, vol. 79, pp. 12-49, 1988.

[14] H. Peng, X. Zhou, F. Li, X. Xia, S. T.C. Wong, "Integrating Multi-Scale Blob/Curvilinear Detector Techniques and Mutli-Level Sets For Automated Segmentation of Stem Cell Images", *IEEE Int Symp Biomed Imaging,* pp. 1362-1365, 2009.

[15] W. K. Pratt, "Digital Image Processing", *Wiley-Interscience*, 4th edition, 2007.

[16] N.Ray, S.T. Acton, "Active Contours for Cell Tracking", *IEEE Southwest Symposium on Image Analysis and Interpretation,* 2002.

[17] V.Shapiro, G.Gluhchev. "Multinational license plate recognition system: segmentation and classification", *IEEE Pattern Recognition Conference,* vol 4., 2004.

[18] J. Simon, (2015, May 03). DGradient. Retrieved from http://www.n-simon.de/mex/.

[19] J. Soukup, P. Císař, F. Šroubek, "Segmentation of Time-Lapse Images with focus on Microscopic Images of Cells", *ICIAP*, 2013.

[20] B. Vemuri, Y. Chen, "Joint Image Registration and Segmentation", *Geometric Level Set Methods in Imaging, Vision and Graphics*, *Springer*, pp. 189-224, 1995.

[21] Y.J. Zhang, "A Survey on Evaluation Methods for Image Segmentation", *Pattern recognition 29 (8),* pp. 1335-1346, 1996.

[22] H. Zhao, T. Chan, B. Merriman, S. Osher, "A variational level set approach to multiphase motion", *J. Comp. Phys.*, vol. 127, pp. 179-195, 1996.