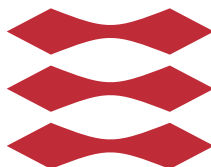


# Segmentation and tracking of neural progenitor cells in microscopic image sequences

John Christian Højland

DTU



Kongens Lyngby 2013  
IMM-M.Sc.-2013-65

Technical University of Denmark  
Informatics and Mathematical Modelling  
Building 321, DK-2800 Kongens Lyngby, Denmark  
Phone +45 45253351, Fax +45 45882673  
[reception@imm.dtu.dk](mailto:reception@imm.dtu.dk)  
[www.imm.dtu.dk](http://www.imm.dtu.dk) IMM-M.Sc.-2013-65

# Summary (English)

---

The goal of the thesis is to present a pipeline for automatic segmenting and tracking of cells in phase-contrast microscopy image sequences. The result from such a pipeline is a number of graphs showing the lineage of the cells from the initial image frame and their development as it progresses over time. The collected data contain information on position, size and speed of each cell and also frequency of mitosis for each cell. This is useful for research in stem cells and the development hereof. This is an essential tool for registering and comparing the effects of different treatments of the cell cultures.

The segmentation and tracking is based on level-set theory and implemented as a modified version of the active contour formulation as proposed by Chan and Vese [CV01]. In the active contour model a level set of an implicit function is used to define the contour. The cells can not merge and therefore the model also needs to be constrained to avoid the individual segments merge. A cost-function is constructed which is minimized through a series of iterations between each frame. The active contour model adapt the contour to the shape and movement of the cell between each frame.

The model is applied to the dataset for validation purposes and for comparison of performance . The dataset used is fully manually annotated and therefore suited for validation and benchmarking.



# Summary (Danish)

---

Målet for denne afhandling er at præsentere en pipeline til automatisk segmentering og tracking af celler i fase-kontrast mikroskopi billed sekvenser. Resultatet af en sådan pipeline er en række grafer der viser slægtskab af cellerne fra det oprindelige billede og dennes udvikling over tid. Fra de opsamlede data kan oplysninger om position, størrelse og hastighed for hver enkelt celle samt mistose frekvensen for hver celle udtrækkes. Dette er brugbart i bla. stamcelle forskningen og dennes videre udvikling. Dette er et nødvendigt værktøj til at registrere og sammenligne forskellige behandlinger af celle kulturene.

Segmenteringen og trackingen er baseret på level-set teori og er implementeret som en modificeret version af active contour model, foreslået og beskrevet af Chan og Vese [CV01]. I active contour model beskrives contouren som et levelset af en implicit funktion. Cellerne kan ikke smelte sammen, så derfor skal modellen også begrænses så den ikke tillader segmenter at smelte sammen. En kost funktion konstrueres og denne minimeres iterativt mellem hvert billede. Active contour model tilpasser sig og følger formen og bevægelsen af den enkelte celle mellem hvert billede.

Modellen afprøves på datasættet, for at validere modellen og for at sammenligne resultatet . Datasættet der bruges til validering er fuldt manuelt annoteret og derfor velegnet til validering og sammenligning af resultater.



# Preface

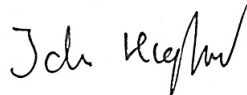
---

This thesis was prepared at the department of Informatics and Mathematical Modelling at the Technical University of Denmark in fulfilment of the requirements for acquiring an M.Sc. in Informatics.

The thesis deals with segmentation and tracking of cells in microscopic image sequences.

The thesis consists of six chapters, followed by an Appendix.

Lyngby, 02-July-2013

A handwritten signature in black ink, reading "John Højland". The signature is written in a cursive style with a large initial 'J' and a long, sweeping underline.

John Christian Højland





# Acknowledgements

---

I would like to thank my supervisors professor Rasmus Larsen, professor Knut Conradsen and Jacob S. Vestergaard for inspiration and constructive feedback throughout the duration of this project.

Furthermore I would like to thank my family for their support and patience.



# Contents

---

<b>Summary (English)</b>	<b>i</b>
<b>Summary (Danish)</b>	<b>iii</b>
<b>Preface</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Data . . . . .	2
1.1.1 Microscopy . . . . .	2
1.1.2 The data sets . . . . .	3
<b>2 Theory</b>	<b>7</b>
2.0.3 Why Level set metod . . . . .	8
2.0.4 A curve as an implicit function . . . . .	11
2.0.5 Cartesian grid . . . . .	13
2.0.6 Interpolation . . . . .	13
2.0.7 Properties of implicit curves . . . . .	14
2.0.8 Signed distance function . . . . .	15
2.1 Levelset method . . . . .	16
2.1.1 Motion involving mean curvature . . . . .	16
2.2 The model . . . . .	18
2.2.1 Divergence . . . . .	20
2.2.2 Heaviside function . . . . .	21
2.2.3 Delta dirac function . . . . .	21

---

<b>3</b>	<b>Implementation</b>	<b>23</b>
3.1	Preprocessing . . . . .	24
3.1.1	Background equalization . . . . .	24
3.2	Segmentation . . . . .	28
3.2.1	Initial segmentation . . . . .	28
3.2.2	Global segmentation . . . . .	29
3.2.3	Local segmentation . . . . .	30
3.2.4	Walk-through . . . . .	31
3.2.5	Mitosis detection . . . . .	35
3.3	Tracking . . . . .	35
3.3.1	Renumbering . . . . .	35
3.3.2	Tracking paths . . . . .	36
3.4	Validation . . . . .	38
3.4.1	Validation of segmentation . . . . .	38
3.4.2	Validation of tracking . . . . .	40
<b>4</b>	<b>Test</b>	<b>41</b>
4.1	Segmentation . . . . .	42
4.1.1	Test 1 . . . . .	42
4.1.2	Test 2 . . . . .	45
4.1.3	Test 3 . . . . .	46
4.1.4	Test 4 . . . . .	47
4.1.5	Validation of segmentation . . . . .	50
4.1.6	Test 5 . . . . .	50
4.1.7	Test 6 . . . . .	51
4.1.8	Test 7 . . . . .	52
4.1.9	Test 8 . . . . .	53
4.2	Results . . . . .	54
4.2.1	segmentation . . . . .	54
4.2.2	Tracking . . . . .	55
<b>5</b>	<b>Conclusion</b>	<b>59</b>
<b>6</b>	<b>Future work</b>	<b>61</b>
<b>7</b>	<b>Appendix1</b>	<b>63</b>
7.1	Structures . . . . .	63
7.1.1	Track_data . . . . .	63
7.1.2	Work matrix . . . . .	65
7.1.3	Save matrix . . . . .	66
	<b>Bibliography</b>	<b>67</b>

# List of Figures

---

1.1	Phase contrast microscopy image of pig neural progenitor cells (subsection).	4
1.2	Phase contrast microscopy image of rat pancreas adult stem cells	4
2.1	A section of a circular interface expanding in the normal direction	8
2.2	A circle with a tangent and gradient	9
2.3	Corner expanding in the normal direction	9
2.4	A plot of the function $\phi(x, y)$ and the levelset $\phi(x, y) = 0$ in red	10
2.5	The implicit function of the unit circle	11
2.6	Three levelsets of the implicit function $\phi$	12
2.7	Bilinear interpolation	14
3.1	Flowchart of the main program	23
3.2	Phase contrast microscopy image of rat pancreas adult stem cells	24
3.3	Manually sampled image	25
3.4	Estimated background intensity	27
3.5	Background corrected image	27
3.6	Flowchart of the segmentation	28
3.7	Global segmentation	29
3.8	Missed object	30
3.9	Detailed flowchart of the segmentation	31
3.10	Initial cellmask	32
3.11	Distance function	32
3.12	Heaviside function	33
3.13	Curvature	34
3.14	Flowchart of the tracking	36
3.15	Flowchart of the segmentation validation	38
3.16	Fully segmented image, timeframe 200	39

---

4.1	Initial contour inside object . . . . .	42
4.2	Initial contour around segments . . . . .	43
4.3	Initial contour outside segments . . . . .	43
4.4	No segment present . . . . .	44
4.5	Initial contour overlapping segment . . . . .	44
4.6	Initial states of test 2 . . . . .	45
4.7	$\phi$ and force evolved over time . . . . .	46
4.8	Initial contour overlapping segment . . . . .	47
4.9	Flowchart of the segmentation validation . . . . .	47
4.10	Test for merge . . . . .	48
4.11	test for split . . . . .	49
4.12	Segmentation Validation Test 5. . . . .	50
4.13	Segmentation Validation Test 6. . . . .	51
4.14	Segmentation Validation Test 7. . . . .	52
4.15	Segmentation Validation Test 8. . . . .	53
4.16	Overlap error . . . . .	54
4.17	Validation problem, timeframe 29 . . . . .	55
4.18	Validation problem . . . . .	56
7.1	Track_ data struct . . . . .	63
7.2	Track_ data.cell . . . . .	64
7.3	True data, Segmentation, overlap and validated with overlap . . . . .	65
7.4	The first 20 rows of the save matrix . . . . .	66

# Introduction

---

Automated microscopy is an increasingly important area of research. Due to the size of datasets from time lapse imaging of in-vitro experiments it is no longer feasible to rely on manual annotation. Extraction of useful information from these image sequences, require robust and reliable algorithms to be developed. A study by [RBM<sup>+</sup>11], provides the community with 2 fully manually annotated data sets, and an automated tracking algorithm which can serve as a benchmark for future work. The cell tracking algorithms can roughly be subdivided into three types of algorithms, according to [MDSvC09] and [PRR10].

- Contour evolution, level sets are one such algorithm.
- Stochastic filtering, where the Kalmanfilter, the Particle filter and mean shift tracking belong.
- Segmentation and association. The Hungarian algorithm is an example of association.

Some of the problems of segmentation and association mentioned in [MDSvC09] and [PRR10] are a low ability to register mitosis, appearance and disappearance of cells which cause changes in topology. This is one of the strengths of the contour evolution and therefore this is the approach that is taken in this thesis.

## 1.1 Data

### 1.1.1 Microscopy

This section is written for those who have no prior knowledge of microscopy. It is a short explanation of the specific techniques used for acquiring the images in the two datasets.

The name microscope come from ancient Greek and roughly translated mean : to see small objects . A microscope is used when reproduction ratios higher than 1:1 is needed. Dependent of the objective used in the microscope reproduction ratios of up to 1500:1 is possible. The lower limit of the resolution, when visual light is used, is 0,2 micrometers and it is caused by diffraction of the light. In order to produce high reproduction ratios the objective lens is placed very close to the specimen, causing the depth of field to be very low. In order to be able to better control the light, an artificial light source is used. The illumination shall be evenly distributed across the specimen, glare free and for most purposes, visual light with a wide dynamic range.

#### 1.1.1.1 Bright-field illumination

Bright-field illumination is the most commonly used illumination in optical microscopy. The light source is usually a tungsten or halogen lamp. The light passes through its own optical system and is aimed at the specimen from underneath. The intensity levels in the resulting image is caused by absorption of the light by the specimen. Very small cells usually can not absorb much light hence the resulting contrast is low. The contrast can be improved by staining the specimen, but the dye might interact with the specimen, especially if it is living cells. This is an unwanted side effect and therefore other noninvasive contrast enhancement techniques are usually applied.

#### 1.1.1.2 Oblique illumination

Oblique illumination or an-axial illumination is obtained by changing the path of the light from the light source so that it passes through the specimen at an angle. The angled light cause the light to be a little brighter on one side of the sample and a shadow to be cast on the opposite side. This give a slightly better modeling of the specimen and a sense of depth to the image. It also enhances the contrast.



### 1.1.1.3 Phase contrast

A lot of living biological specimens have no color and are so small that they are nearly transparent. Normally an image of such specimen will have very low contrast. The contrast can be enhanced by a phase contrast objective. The refraction index is different in the background substance than in the cell. This cause the light to take slightly different paths through the sample. The resulting light intensity is almost the same but the phase contrast objective can see the difference in angle and it applies a gray-filter to lower the intensities from the light that travels in one direction and leave the rest intact. This increase the contrast of the image but also typically introduce a halo effect to the image.

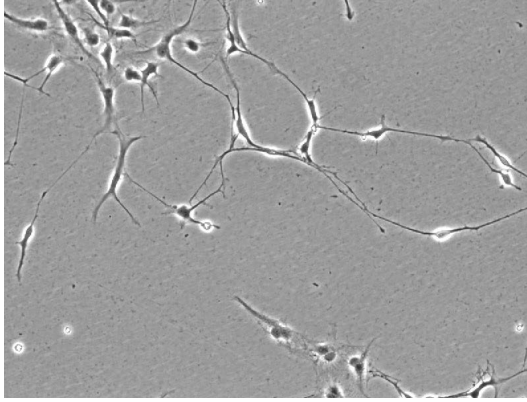
## 1.1.2 The data sets

The data provided for this project consist of two separate datasets. The images of both datasets are acquired using a phase contrast objective lens but are quite different. The phase contrast objective lens is used for several reasons. One reason is that adding chemicals such as florescence to the sample is not wanted, since it may interact with the sample. Another reason is that it enhances contrast in the image especially around the edges of the cells.

The first data set contains image sequences of neural progenitor stem cells from a pig. The images are obtained from a phase contrast microscope and each frame is divided into 16 sub areas which are stored separately. The sequence contain 1272 images acquired every 5 minutes over a total duration of 106 hours. The raw data consists of 8-bit gray-scale images with a size of  $3200 \times 2400$  pixels. Bright field illumination is used. A zoomed in example of one of the images is seen in figure 1.1

In the image it can be seen that the cells vary a lot in shape and size. Phase contrast microscopy typically add a halo around the cell which is seen as bright white areas around the cell body. Neural cells form networks by attaching to each other with axons, making it much harder to separate each cell in the segmentation. The cells change shape as they move around and they multiply by mitosis, which is causing a change in typology and is an issue that has to be addressed. Another challenge is the fact that some cells, as they move around disappear from the field of view while others appear.

The other dataset consist of images from a project performed by the university of



**Figure 1.1:** Phase contrast microscopy image of pig neural progenitor cells (subsection).

Lbeck and is made publicly available. This dataset is fully manually annotated and therefore is well suited for validation and benchmarking. The images show adult stem cells from the pancreas of a rat. Images were acquired using a phase-contrast objective lens. The raw image data consist of 12bit gray scale images with a size of  $1376 \times 1038$  pixels recorded every 15 minutes (refdataA) over a total of 52 hours. Oblique illumination is used. An example of the images can be seen in Figure 1.2



**Figure 1.2:** Phase contrast microscopy image of rat pancreas adult stem cells

From this image it can be seen that the background is unevenly lit, which have to be taken care of in a preprocessing. The cells compared to the other dataset are more similar to each other and have a more even shape. They also seem less attached to each other. All in all this dataset seems to be an much easier task

and therefore better suited to start with. Together with the fact that the dataset is fully manually annotated makes it the obvious choice for the implementation and verification of the automatic segmentation. Later on when the algorithm is verified and fully functional it can be applied to the other dataset.



## CHAPTER 2

# Theory

---

This section is primarily based on theory from the book *Level Set Methods and Dynamic Implicit Functions*[OF03] and from the scientific paper [CV01]. The theory presented is merged from the sources and ordered to explain the theoretical model later implemented in this thesis. Only the theoretical concepts used in the implementation are covered.

### 2.0.3 Why Level set method

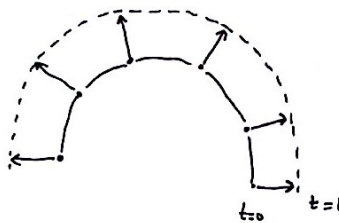
As stated in the introduction the cell tracking algorithms can roughly be subdivided into three types of algorithms, according to [MDSvC09] and [PRR10].

- Contour evolution, active contours are one such algorithm.
- Stochastic filtering, where the Kalmanfilter, the Particle filter and mean shift tracking belong.
- Segmentation and association. The Hungarian algorithm is an example of association.

Some of the problems of segmentation and association mentioned in [MDSvC09] and [PRR10] are a low ability to register mitosis, appearance and disappearance of cells which cause changes in topology. This is one of the strengths of the contour evolution and therefore this is the approach that is taken in this thesis.

#### 2.0.3.1 Motion in normal direction

If we initially start with a contour that is shaped as a circle and see how this evolves. A natural direction the circle would evolve would be in the direction of the gradient. The positive gradient direction will make the circle larger and the negative gradient direction will make the circle smaller.

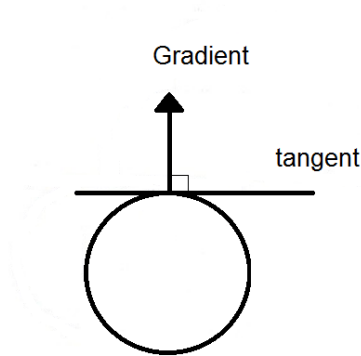


**Figure 2.1:** A section of a circular interface expanding in the normal direction

The gradient of the circle at a specific point on the contour  $\phi(x, y)$  is defined as:

$$\text{grad } \phi = \nabla \phi = \left( \frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y} \right) \quad (2.1)$$

This is a vector in the normal direction, which is perpendicular to the surface of  $\phi$ , and in the xy plane this is perpendicular to the tangent direction and hence perpendicular to the isocontour.

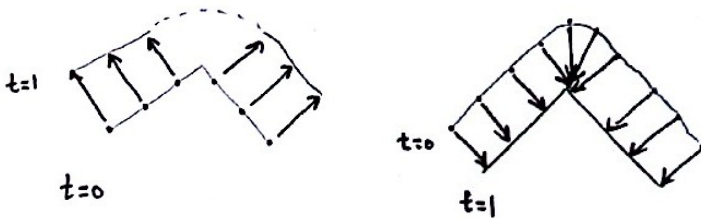


**Figure 2.2:** A circle with a tangent and gradient

To get the unit normal vector ( $\vec{N}$ ) we simply divide by the length of the gradient:

$$\vec{N} = \frac{\nabla\phi}{|\nabla\phi|} \quad (2.2)$$

If formulated as particles the particles will spread out and get further apart as a function of time or the opposite. This is not desirable when solving the problem numerically. If the points are too close together it is not possible to calculate a differentiation and if they are too far apart it is very imprecise when interpolating the contour.



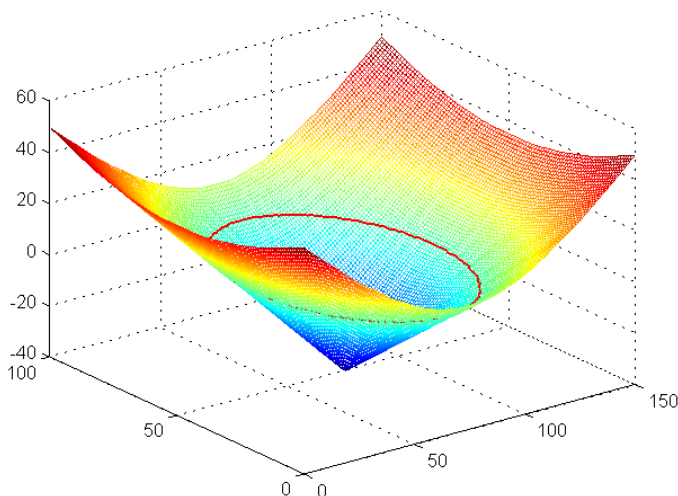
**Figure 2.3:** Corner expanding in the normal direction

As seen in Figure 2.3 on the left: expanding a corner leads to uneven sample points. On the right: a smooth contour can evolve to be a sharp corner. Another problem in the particle approach is finding out if a pixel is inside or outside the contour.

Another approach is to use a level set method. A level describe a function at a certain function value:

$$f(\phi) = c \quad (2.3)$$

A levelset is then a set of these solutions, which you can imagine stacked on top of each other.



**Figure 2.4:** A plot of the function  $\phi(x, y)$  and the levelset  $\phi(x, y) = 0$  in red

Moving up one dimension from 2D to 3D by displaying the contour as a function of time enable us to describe the evolution of our 2D contour in this way. As every time step is constant (unit length) the time and distance can replace each other which is one reason why the distance function is a good choice for describing the level set. This provide us with a continuous function that always is negative at areas inside the contour always positive at areas outside the contour and zero at the contour. Now making the segmentation is quite easy done by including the level set of the contour to the inside area and then defining the background as  $f(\phi) > 0$  and the foreground as  $f(\phi) \leq 0$

The evolution of the curve defined by the distance function  $\phi$  at a given speed  $F$  is described by the differential equation:

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| F, \phi(0, x, y) = \phi_0(x, y) \quad (2.4)$$

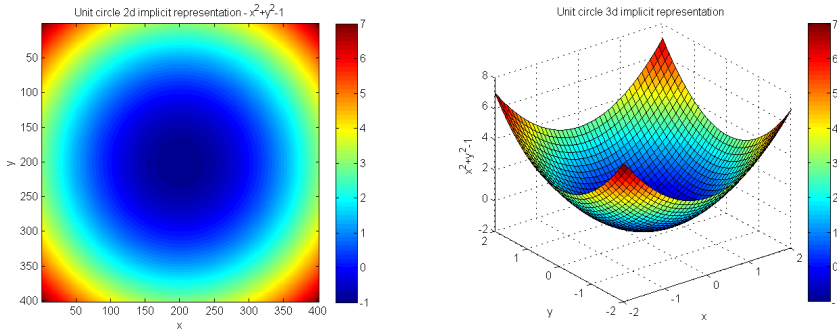


## 2.0.4 A curve as an implicit function

When working in two dimensional space, the interface is a curve that separates the the domain into two areas with nonzero values. It is necessary to constrain the problem to only address closed curves, but in our case this is later taken care off by the Mumford-Shah formulation. As an example the unit circle can be described as a implicit curve.

$$\phi(x, y) = x^2 + y^2 - 1 \quad (2.5)$$

which constitutes the three dimensional implicit function is illustrated in Figure 2.5



**Figure 2.5:** The implicit function of the unit circle

and by choosing a set equal to zero:

$$x^2 + y^2 - 1 = 0 \quad (2.6)$$

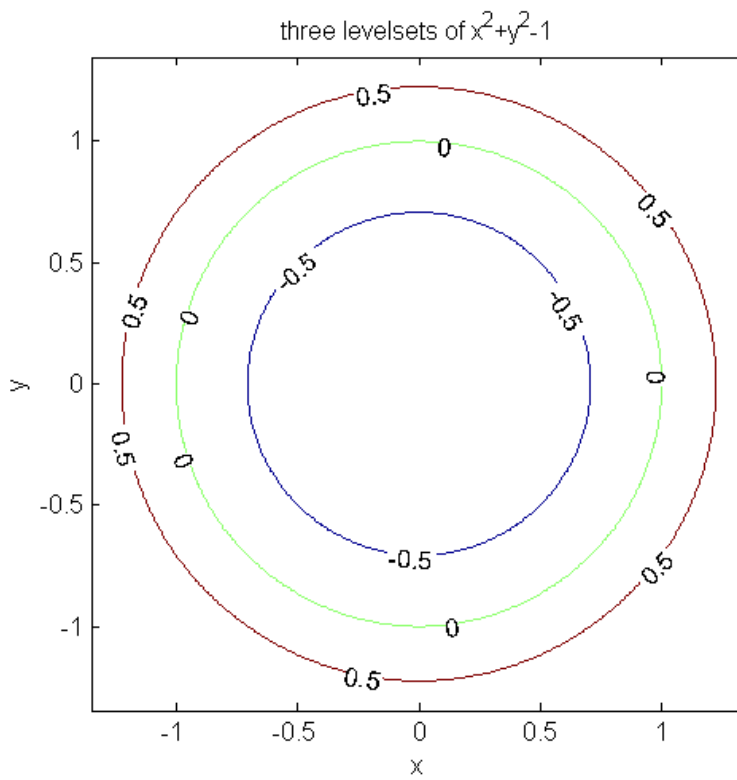
The resulting isocontour is the unit circle. This can be seen in Figure 2.6 where the unit circle is plotted together with circles where  $x^2 + y^2 - 1 = c$  and  $c = -0.5$  and  $0.5$ .

To establish a notation for the regions, consider  $\phi(\vec{x}) = x^2 + y^2 - 1$  where the interface is defined by the  $\phi(\vec{x}) = 0$  isocontour

The interior region  $\Omega^-$  is defined as  $\Omega^- = \{\phi(\vec{x}) \mid \|\phi(\vec{x})\| < 1\}$ .

The exterior region  $\Omega^+$  is defined as  $\Omega^+ = \{\phi(\vec{x}) \mid \|\phi(\vec{x})\| > 1\}$ .

The interface  $\partial\Omega$  is defined as  $\Omega = \{\phi(\vec{x}) \mid \|\phi(\vec{x})\| = 1\}$ .



**Figure 2.6:** Three levelsets of the implicit function  $\phi$

## 2.0.5 Cartesian grid

When implementing the algorithm the representation have to be discretized, the location of the interface is not given by neither the explicit or the implicit discrete representation. The discretized representation only hold information at a finite number of sample points and all values in between has to be determined by interpolation. There are standard procedures for interpolating the contours in most software packages. The logical choice for how to sample, when dealing with images is a Cartesian grid which is defined as:

$$\{(x_i, y_j) \mid 1 \leq i \leq m, 1 \leq j \leq n\} \quad (2.7)$$

The grid are chosen to be uniform in the way that it is equidistantly sampled and the scale is the same in both x and y direction. Having the same scale in both direction cause the approximation errors to be the same in both direction. The Cartesian grid at the same time impose a rectangular domain defined by:

$$D = [x_1, x_m] \times [y_1, y_n] \quad (2.8)$$

It is obvious that only points close to the interface is needed for the interpolation, while points far from the interface does not have to be calculated. This also reduce the computational power and memory usage needed to perform the calculations without compromising the result.

## 2.0.6 Interpolation

The calculations in the model is based on differentials calculated from central distance approximations, which are linear approximations. If any interpolation has to be made on these data a bilinear interpolation is sufficient.

**Bilinear interpolation** Linear interpolation use a straight line to connect each pixel value. In 2D the linear interpolation is called Bilinear. In the bilinear interpolation the weighted average of the distances to the four nearest pixel is assigned to the output image.  $\hat{I}(\hat{x}, \hat{y})$  is the interpolated value in the output image. See figure 2.7 for the entities  $dx, dy, I_{00}, I_{01}, I_{10}$  and  $I_{11}$

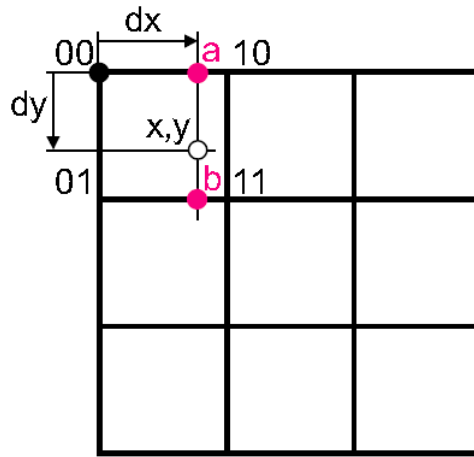


Figure 2.7: Bilinear interpolation

Then the linear interpolation can be computed as :

$$\hat{I} = I_{00}(1 - dx)(1 - dy) + I_{10}dx(1 - dy) + I_{01}(1 - dx)dy + I_{11}dxdy \quad (2.9)$$

### 2.0.7 Properties of implicit curves

The implicit representation has some very powerful properties. The interface is defined such that the isocontour where  $\phi(x, y) = 0$ , the interior region  $\Omega^- = \phi(x, y) < 0$  and the exterior region  $\Omega^+ = \phi(x, y) > 0$  which make it easy to determine if any point is inside or outside the interface, simply by determining the sign of  $\phi$ .

Numerical interpolation introduce approximation errors in the determination of  $\phi$ . This could cause an inside point to be determines as outside or the opposite. This may sound destructive for the algorithm, but in fact this only cause the interface to be moved slightly. If the errors are small, as is assumed in most numerical methods, these errors are minor and can be accepted. From calculus we know that the errors can be decreased by increasing the number of samples, but this is not a feasible solution as the size of the problem get larger. A better approach is to enforce a certain level of smoothness, which make it possible to numerically approximate the solution with smaller error.

## 2.0.8 Signed distance function

The function  $\phi$  is preferred to be smooth when performing numerical approximations or sampling by interpolation. There should not be steep or flat gradients nor any kinks. This can be accommodated by the use of the signed distance function. The signed distance function accommodates all the needed criterion for an implicit function and in it self it is a subset of implicit functions. The signed distance function is defined so the exterior region is positive, the interior region is negative and the interface is zero. An extra condition is imposed to ensure the step length in the xy-plane is constant and equal to one:

$$\|\nabla\phi(\vec{x})\| = 1 \quad (2.10)$$

A distance function is defined as :

$$d(\vec{x}) = \min(\|\vec{x} - \vec{x}_I\|) , \forall \vec{x}_I \in \partial\Omega \quad (2.11)$$

This imply that  $d(\vec{x}) = 0$  on the boundary where  $(\vec{x}) \in \partial\Omega$ . For all other points the shortest distance can be decided by the steepest decent, which is the gradient direction. Since  $d$  is the Euclidean distance:

$$\|\nabla d\| = 1 \quad (2.12)$$

This is in general true, but not if the point is equidistant between two closest points on the interface. This is one of the imperfections of the distance function. However these points are usually not close to the interface. Another imperfection is a local extrema on the interface where  $d = 0$  that will cause a kink in the distance function, which again will make it difficult to approximate the derivatives near the interface at this point.

Determining the closest point, amount to evaluating a point  $\vec{x}$  on the Cartesian grid and the fact that  $\phi(\vec{x})$  is the signed distance function together with the gradient which point in the direction of the closest point  $\vec{x}_C$  on the interface, it can be determined by:

$$\vec{x}_C = \vec{x} - \phi(\vec{x})\vec{N} \quad (2.13)$$

## 2.1 Levelset method

### 2.1.1 Motion involving mean curvature

The interface should be able to evolve and adapt to shapes in an image and in order to do that a velocity field is necessary. An internal velocity field  $\vec{V}$  is imposed which depend on the implicit function  $\phi$ . Motion by mean curvature is when the interface moves in the normal direction with a velocity proportional to its curvature:

$$\vec{V} = -b\kappa\vec{N} \quad (2.14)$$

where  $b$  is a constant and  $\kappa$  is the curvature for a two-dimensional curve given implicitly by  $f(x, y) = 0$  is defined by:

$$\kappa = \frac{f_{xx}f_y^2 - 2f_{xy}f_xf_y + f_{yy}f_x^2}{(f_x^2 + f_y^2)^{\frac{3}{2}}} \quad (2.15)$$

When  $b \geq 0$  the interface moves in the direction of concavity, which will make a circle shrink to a point and ultimately vanish.

The motion by mean curvature is characterized by

$$V_n = -b\kappa \quad (2.16)$$

The velocity field for motion by mean curvature points in the normal direction and therefore there is not tangential component. The levelset equation look like this:

$$\phi_t + V_n\vec{N} \cdot \nabla\phi = 0, \quad (2.17)$$

where  $V_n$  is the velocity in the normal direction also known as the *normal velocity*. Furthermore the gradient point in the normal direction so the unit normal cancel out in the levelset equation:

$$\vec{N} \cdot \nabla\phi = \frac{\nabla\phi}{\|\nabla\phi\|} \cdot \nabla\phi = \frac{\|\nabla\phi\|^2}{\|\nabla\phi\|} = \|\nabla\phi\| \quad (2.18)$$

The levelset equation reduces to:

$$\phi_t + V_n \cdot \|\nabla\phi\| = 0, \quad (2.19)$$

And after plugging in the motion by mean curvature,

$$\phi_t - b\kappa\|\nabla\phi\| = 0, \quad (2.20)$$

and rearranging

$$\phi_t = b\kappa\|\nabla\phi\|. \quad (2.21)$$

## 2.2 The model

The model that is used in this thesis is based on an active contour model without edges suggested by Chan and Vese [CV01]. The Mumford Shah approach to solving the minimal partition problem is used to create the stopping criterion.

The model in its simplest form need an initial contour, a term that will evolve the contour in the normal direction and a stopping criterion based on the image intensities. The model is implemented as a global energy minimization. This is explained in[Ran05] The first term is only based on the initial contour it self and not the image intensities. This term is called the internal energy. The stopping criterion is based on the image intensities, and is also called the external energy. The model evolves the contour through the spacial domain of the image and minimizes the functional of energy:

$$E_{total} = E_{internal} + E_{external} \quad (2.22)$$

Each term is assigned a scaling factor that determine their mutual weight.

$$E_{total} = \mu \cdot E_{internal} + \lambda \cdot E_{external} \quad (2.23)$$

as explained in section 2.0.3, the evolving term is a motion by mean curvature, evolving the contour in its gradient direction and the evolution is curvature dependent, thus the curvature and the unit normal is needed. These are all entities based on the gradient of the function  $\phi$ :

The term that stops the evolution is deduced from image features. In the classical snake model, it is based on an edge detection. This has the disadvantages: it is sensitive to noise, there might be "holes" in the edge and the edges might not have the same "strength" all around the blob. Instead I use a segmentation based on the Mumford Shah functional.

Now in order to explain the stopping criterion  $E_{external}$  let us only focus on the energy of the the stopping criterion  $E_{external}$  as in Chan Vese [CV01]. In the following some definitions are used:

$\partial\Omega$  is the evolving curve in  $\Omega$  that form a boundary around an open subset  $\omega$  ( $\omega \subset \Omega, \partial\Omega = \partial\omega$ )

The inside region is denoted  $\omega^-$

The outside region is denoted  $\omega^+$



When the model is applied to an image  $u_0$ . The image is formed by two piecewise constant regions. Then if we consider the fitting term:

$$\begin{aligned}
 F_1(\partial\omega) + F_2(\partial\omega) = & +\lambda_1 \int_{\omega^-} |u_0(x, y) - c_1|^2 dx dy \\
 & +\lambda_2 \int_{\omega^+} |u_0(x, y) - c_2|^2 dx dy
 \end{aligned} \tag{2.24}$$

It is obvious that if the constants  $c_1$  and  $c_2$  are taken to be the mean values of the region  $\omega^-$  and respectively  $\omega^+$  the minimizer

$$\inf_{\partial\omega} F_1(\partial\omega) + F_2(\partial\omega) \approx 0 \approx F_1(\partial\omega_0) + F_2(\partial\omega_0) \tag{2.25}$$

is on the boundary of the object in the image.

With the regularizing term added the model look like this:

$$\begin{aligned}
 F(c_1, c_2, \partial\omega) = & \mu \cdot Length(\partial\omega) \\
 & +\lambda_1 \int_{\omega^-} |u_0(x, y) - c_1|^2 dx dy \\
 & +\lambda_2 \int_{\omega^+} |u_0(x, y) - c_2|^2 dx dy
 \end{aligned} \tag{2.26}$$

The equation 2.26 is equivalent to the Mumford-Shah function for segmentation as in [MS89]:

$$\begin{aligned}
 F^{MS}(u, \partial\omega) = & \mu \cdot Length(\partial\omega) \\
 & +\lambda \int_{\omega^-} |u_0(x, y) - u(x, y)|^2 dx dy \\
 & + \int_{\omega^+} |\nabla u(x, y)|^2 dx dy
 \end{aligned} \tag{2.27}$$

### 2.2.1 Divergence

The divergence of a vector field  $div(F)$  can also be written  $\nabla \cdot F$ , where the nabla operator give the partial derivative as in the gradient :

$$\nabla = \left( \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right) \quad (2.28)$$

The divergence is the dot product of the nabla operator and the vector field  $F$ , resulting in :

$$div F = \nabla \cdot F = \left( \frac{\partial F}{\partial x}, \frac{\partial F}{\partial y} \right) \quad (2.29)$$

In the active contour model the divergence is used to calculate the mean curvature. The mean curvature is the divergence of the unit normal vectors along the curve. As in :

$$\nabla \cdot \vec{N} = \nabla \cdot \frac{\nabla \phi}{\|\nabla \phi\|} \quad (2.30)$$

The divergence come from the conservation laws in physics. The two dimensional version of the divergence theorem is known as Green's theorem in a plane and according to [KMS02] is defined as:

$$\oint_C (Pdx + Qdy) = \iint_R \left( \frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} \right) \quad (2.31)$$

Where  $C$  is a contour enclosing a region  $R$ .  $P(x, y)$  and  $Q(x, y)$  are function that are single valued, finite and continuous inside and on the boundary of  $R$ . Green's theorem in the plane links the surface integral along  $C$  to the double integral over the entire area  $R$ .

In another form it is known as Gauss's law which relate the integral of the divergence to the flux:

$$\iint_R \left( \frac{\partial \phi}{\partial x} - \frac{\partial \phi}{\partial y} \right) dx dy = \oint W \cdot n ds \quad (2.32)$$

Where  $W$  is the flux and  $n$  is the outward unit normal direction.

### 2.2.2 Heaviside function

The Heaviside function is used to do calculation on an area, and is defined as:

$$H(\phi) = \begin{cases} 1, & \text{if } \phi \leq 0 \\ 0, & \text{if } \phi > 0 \end{cases} \quad (2.33)$$

It adds the contour to the inside of the enclosed region.

### 2.2.3 Delta dirac function

The delta dirac function is used to pick out the contour, from the Heaviside function as :

$$\delta_0(\phi) = \frac{d}{d\phi} H(\phi) \quad (2.34)$$

Having the Heaviside function and the Delta dirac function enable the calculation of a line as a surface integral of a quantity  $p(x, t)$  over the contour  $\partial\omega$  as:

$$\int_R p(x, t) \delta(\phi) |\nabla \phi| dx. \quad (2.35)$$

And also calculation of the area integral of  $p(x, t)$  over  $\Omega$  as:

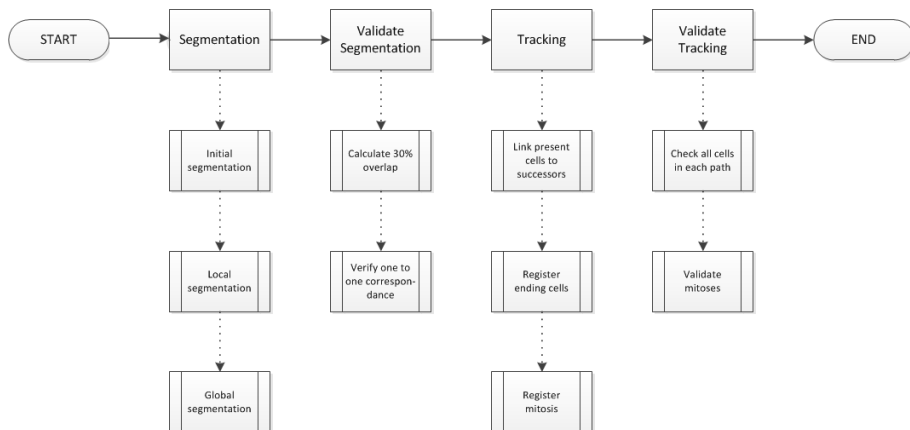
$$\int_R p(x, t) H(\phi) dx. \quad (2.36)$$

This all lead us to the definition of the active contour model used in this thesis:

$$\frac{\partial \phi}{\partial t} = \delta(\phi) \left[ \mu \operatorname{div} \left( \frac{\nabla \phi}{|\nabla \phi|} \right) - \lambda_1 (u_0 - c_1)^2 + \lambda_2 (u_0 - c_2)^2 \right] \quad (2.37)$$

# Implementation

---



**Figure 3.1:** Flowchart of the main program

## 3.1 Preprocessing

The preprocessing is necessary to bring the images into a state where they can be segmented properly. The aim is to separate the cells as foreground and the the rest of the image as background. From the image in Figure 3.2 it is seen that the background have a varying intensity. It is very light in the top right side and quite dark in the bottom corners. This has to be corrected for, if a successful segmentation have to be performed. The reason is that some of the cells in the darker areas have a lower intensity than the background in some parts of the lighter areas, which make it impossible to separate the foreground from the background.

### 3.1.1 Background equalization



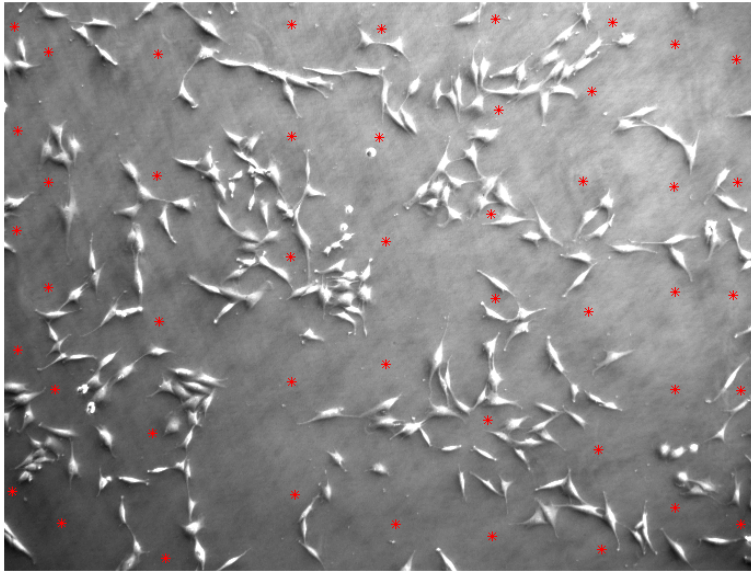
**Figure 3.2:** Phase contrast microscopy image of rat pancreas adult stem cells

The image in figure 3.2 has an uneven illumination across the image, especially the corners are darker. The segmentation is based on intensities and therefore the illumination has to be corrected. Since the background is fairly even, samples of the background can be used to construct a linear regression model, describing the variation in the background as in [?]. When a model is constructed, the correction for each pixel value can be calculated and subtracted from the original

image. The linear regression model take the form:

$$f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j. \quad (3.1)$$

The input vector  $X = (X_1, X_2)$  consist of the manually sampled points from the background in the image as seen in Figure 3.3



**Figure 3.3:** Manually sampled image

The nature of light is that it drops off by the square of the distance and therefore a polynomial quadratic model in both dimensions is chosen. The  $X$  vector is expanded by the basis expansions:  $(X_1, X_2, X_1^2, X_1 \cdot X_2, X_2^2)$  to make it a quadratic function in both dimensions. This give 5 parameters to the model, plus 1 for calculating the intercept. The parameters of the model is solved as a least squares problem. The parameters are minimizing the residual sum squared

of the samples in  $X$ :

$$\begin{aligned}
 RSS(\beta) &= \sum_{i=1}^N (y_i - f(x_i))^2 \\
 &= \sum_{i=1}^N (y_i - \beta_0 - \sum_{j=1}^p x_{ij}\beta_j)^2
 \end{aligned} \tag{3.2}$$

In vector notation the equation take the form:

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta). \tag{3.3}$$

This is a quadratic function which is differentiated with respect to  $\beta$  and set equal to zero to obtain the unique solution:

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X} \mathbf{y} \tag{3.4}$$

Now the estimated background correction can be calculated for each pixel position in the image. Every pixel position is listed in the  $X$  vector, the basis expansions are calculated and then:

$$\hat{y} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X} \mathbf{y} \tag{3.5}$$

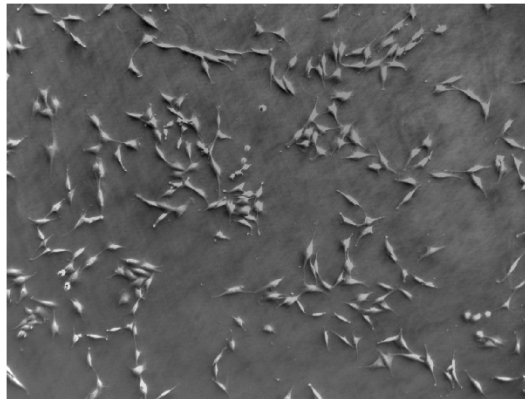


The estimated background correction look like this:



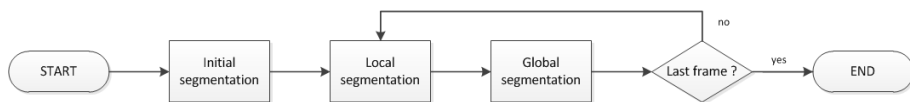
**Figure 3.4:** Estimated background intensity

The original image is corrected by subtracting the background correction from it and the intensities are corrected by a histogram stretch:



**Figure 3.5:** Background corrected image

## 3.2 Segmentation



**Figure 3.6:** Flowchart of the segmentation

Figure 3.6 show a simplified flowchart of the segmentation. The segmentation have to take the fact that cells do not merge into account and this is done by evolving the contour with respect only to the local neighborhood of the contour. This disables the contour from segmenting other objects. Especially the new cells that may enter through the image boundary will not be segmented, so a global segmentation needs to be performed after the local segmentation is finished evolving.

### 3.2.1 Initial segmentation

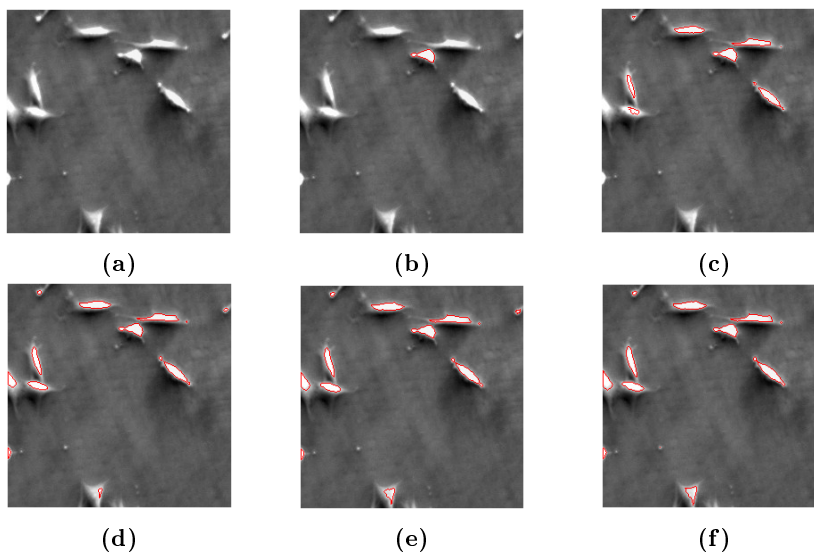
The active contour model evolves a contour over time hence the model needs to be initialized with a contour or a cell mask i the first time frame. This can be initialized in different ways each with different pros and cons:

- The active contour model with one level set can be used to make a global initial segmentation. This make a full automatic segmentation possible. The down side is that touching cells are segmented as one, which introduce errors to the model from the start.
- Manual segmentation of the initial frame. Human interaction is what we try to limit. Now it amounts to manually segmenting only one frame to assure a good quality.
- A combination of both the global segmentation and human intervention to correct the for possible errors, which will limit the time spend on the manual annotation but still ensure the quality.

I chose to simulate the manual annotation by initializing the first frame with the first cell mask from the manually annotated validation data. This is to ensure a good starting point.

### 3.2.2 Global segmentation

The global segmentation can be used to produce the initial segmentation, if wanted. The global segmentation also need an initializer but this could be automated to pick a small neighborhood with intensities above a threshold, since we know the foreground is lighter than the background. A subsection of the size  $250 \times 250$  pixels of the dataset is used for testing

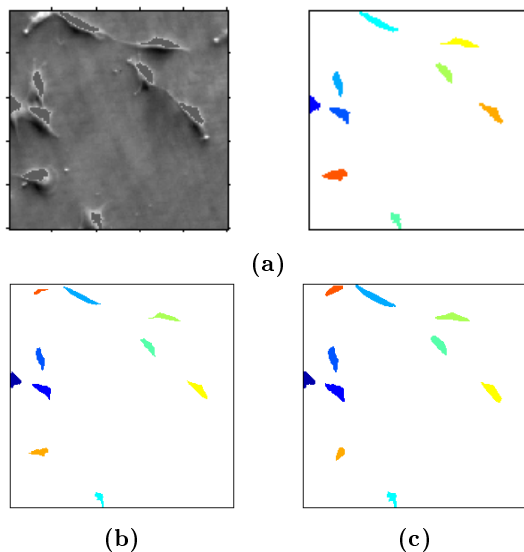


**Figure 3.7:** Global segmentation

As seen in Figure 3.7 the global segmentation work fine as long as the objects are separable. This is a good segmentation of the first frame and a good initializer for the next frame

The local segmentation keep the individual level sets from merging, but also make it impossible to segment any new appearing cells from being segmented. This is why the global segmentation is also needed.

Figure 3.8 top left is the image with the segmented objects grayed out. There is a light object in the top left that has appeared from the border which is not segmented by the local segmentation as seen in the top right image. The bottom left image is after the global segmentation and the object is now segmented and evolved as seen in the bottom right image.



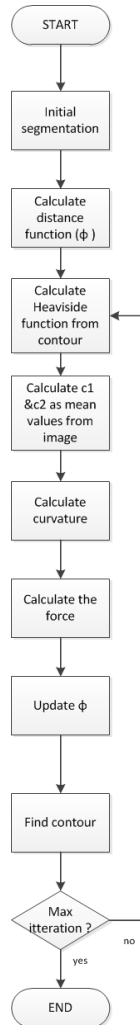
**Figure 3.8:** Missed object

The global segmentation is performed after the local segmentation is fully completed. Now it is assumed that the remaining cells that are not segmented are new cells. These cells are segmented by the global segmentation and separated into individual level sets which are renumbered and added to the local segmentation.

### 3.2.3 Local segmentation

The local segmentation is necessary to keep the individual contours from merging. The local segmentation begin by taking a initial cell mask, which is split into separate level sets. This would take up a lot of memory and be very time consuming if it was done in the naive way. Instead there is only one level set but each contour is identifiable from the cell mask and a larger mask is made by dilation of the initial cell masks. Now each local level set can be developed individually while all other level sets are masked out. The level set is still updated globally which is much faster and use less memory.

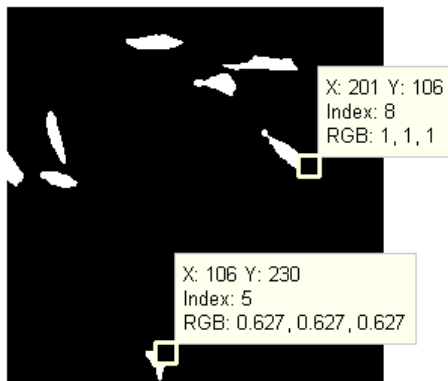
## 3.2.4 Walk-through



**Figure 3.9:** Detailed flowchart of the segmentation

The flowchart in Figure 3.12 show the steps the segmentation go through between each frame. Each step is explained in the following. In order to segment an entire image sequence these steps have to be performed once for each image frame.

The Global segmentation is implemented as an iterative function that can be called from a main script. It takes an initializer in the form of a cell mask as input. The cell mask is at time  $t = 0$ .



**Figure 3.10:** Initial cellmask

Each object in the cell mask is numbered individually as seen in Figure 3.10. From this cell mask the distance function is calculated. This is initially used as the function  $\phi$ .



**Figure 3.11:** Distance function

The distance function is zero where the contour is, positive on the outside and negative on the inside of the contour. The contour is the result of minimizing the cost function.

If the  $\phi$  function does not change, the contour will stay in the same position. To evolve the contour the  $\phi$  has to be changed. This is done iteratively by adding a force to the  $\phi$  function.

This is where the iterations begin. To calculate the force several things are needed. The Heaviside function is calculated. The Heaviside function is used to determine which pixels are inside and which are outside the contour. This is used to calculate the constants  $c_1$  and  $c_2$  in equation 2.26. These constants are the mean value of the foreground and the mean value of the background of an image at time  $t = 1$ . The contour itself is added to the set  $\omega^-$  as in:

$$H(\phi) = \begin{cases} 1, & \text{if } \phi \leq 0 \\ 0, & \text{if } \phi > 0 \end{cases} \quad (3.6)$$

The Heaviside function is similar to the initial cell mask, but is updated each iteration, so it is necessary.



**Figure 3.12:** Heaviside function

Now as in Equation 2.15 the curvature is calculated as:

$$\kappa = \frac{f_{xx}f_y^2 - 2f_{xy}f_xf_y + f_{yy}f_x^2}{(f_x^2 + f_y^2)^{\frac{3}{2}}} \quad (3.7)$$

### 3.2.4.1 Numerical differentiation

On the Cartesian grid the derivatives can be approximated by the first order accurate forward difference:

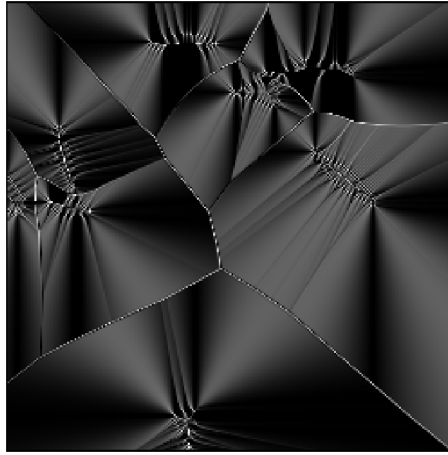
$$\frac{\partial \phi}{\partial x} \approx \frac{\phi_{i+1} - \phi_i}{\Delta x} \quad (3.8)$$

as the first order accurate backward difference:

$$\frac{\partial \phi}{\partial x} \approx \frac{\phi_i - \phi_{i-1}}{\Delta x} \quad (3.9)$$

or as the second order accurate central difference:

$$\frac{\partial \phi}{\partial x} \approx \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x} \quad (3.10)$$



**Figure 3.13:** Curvature

From Figure 3.13 it is seen that the curvature has the highest values near the contours they get lower and more smooth further from the contours. The reason the curvature is not smooth is because the resolution of the image. The  $\kappa$  is very expensive to calculate and since it is a part of the iterations it is performed many times. There is much to gain from optimizing the  $\kappa$  calculation. Now the force can be calculated from Equation :

$$force = \mu \cdot \kappa - \lambda_1(u(x, y) - c_1)^2 + \lambda_2(u(x, y) - c_2)^2 \quad (3.11)$$



To see an example of the updates of the force, look at Test 2, Figure 4.6 Finally the force is normalized by dividing by the maximum absolute value of the force and then the step length  $dt$  is used as a multiplier to determine the weight put on the force as in the update of  $\phi$ :

$$\phi_{i+1} = \phi_i + dt \cdot force \quad (3.12)$$

The update of the  $\phi$  function is different for the local and global segmentations. In the global segmentation all previously known cells are masked out, so it can only find new appearing cells, where in the local segmentation each cell by turn is updated while all others are masked out. The iterations are performed until the stopping criterion is met and the individual segments are evaluated if it still exist and have an area larger than 50 pixels it is kept and otherwise it is deleted as noise or debris. The cells are renumbered and for the tracking later it is registered which cells are mitotic, vanished and which new cells appeared. The successor of each cell is also registered. These are the cells that features in the initializer as well as in the final cell mask.

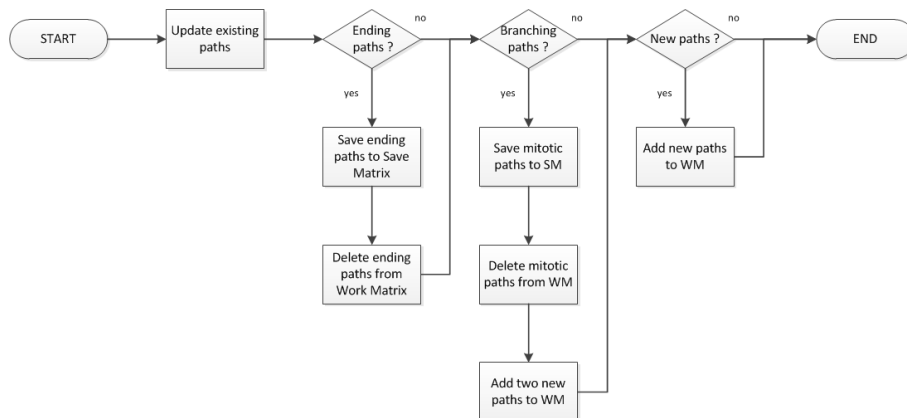
### 3.2.5 Mitosis detection

The mitosis detection can only be reliable if the model is restricted so that the only way it can split is in case of a mitosis. First the contour is divided into one individual contour for each cell. Then this contour is restricted by parameters and by a mask that keep it from segmenting new appearing cells that come into the image frame. The mask is the contour dilated 2 pixels, which is enough to ensure the contour to develop freely locally within the mask, but not develop outside the mask. In this manner the only way the contour can divide into two is if it happens from within the mask and only the mitosis does that.

## 3.3 Tracking

### 3.3.1 Renumbering

Each cell has to be individually identifiable. This can be achieved by assigning an individual number to each cell in any frame ie. the number you end with in the first frame, you continue from in the next frame. Another way of numbering is to assign an individual number to each cell within each frame ie. each frame



**Figure 3.14:** Flowchart of the tracking

is numbered individually, each starting from the number one. In the latter the frame number together with the cell number is a unique identifier for each cell, but the number has the extra information on which frame the cell is present, this is the chosen numbering system. In the first cell mask each cell is numbered from left to right. Each cell mask serve as the initializer for the next frame and the contours are evolved through the iterations. If a cell is present both in the initial cell mask and the resulting cell mask from the segmentation, it continues to the next frame. If a cell vanishes between frames the cell mask will be all zero and it is deleted, but it is registered so it is possible to detect the track is ending in the tracking procedure later. Also if a cells area fall below a threshold of 50 pixels it is considered to be debris and is deleted, these are ignored. The cells that are present both in the initializing cell mask and the final cell mask are renumbered in ascending order beginning from the lowest present cell number. For the tracking these cells are linked to their renumbered successor.

### 3.3.2 Tracking paths

There are two states of a cell, it could be an already known cell, from the previous frame or it could be a new appearing cell. The new cell is either appearing from outside the frame or from through mitosis. The cells can continue to the next frame or they can end, either from mitosis or from going outside the frame.

The path tracking does not need the images any more, but the needed information is taken from the Save Matrix (SM). The Save Matrix is in fact a struct where all the informations from the segmentation is saved. See Appendix 1 for

details.

### 3.3.2.1 Known paths

The paths that existed in the previous frame and still exist are saved to the work matrix. These cells have status = 1. This is performed simultaneously for all entries in a vectorized code.

### 3.3.2.2 Ending paths

The paths that existed in the previous frame but has disappeared have a status = 512. They are saved to the Save Matrix with start frame, end frame, cell numbers and with children = 0. The cells are deleted from the Work Matrix. Whenever a column is deleted from the work matrix, since the cell number is the index the remaining cells are automatically renumbered.

### 3.3.2.3 Branching paths

The paths that have a status = 2 are mitotic. They are saved to the Save Matrix with start frame, end frame, cell numbers and with children as the numbers from the successor vector. After saving the paths they are deleted from the Work Matrix and the two new paths are added at their respective place.

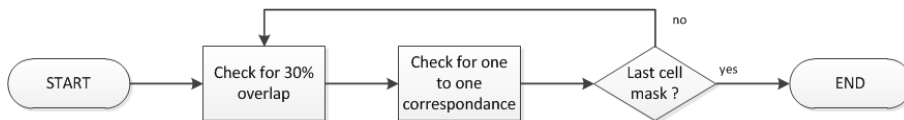
### 3.3.2.4 New paths

Whenever a new path appear it is added to the Work Matrix

## 3.4 Validation

Validation is evaluating the quality of a result. The best way to validate a result is to compare against a know true solution. This will give an exact knowledge to the performance of the algorithm. Such a solution is freely provided by [RBM<sup>+</sup>11] along with the dataset and the result of the performance of their algorithm for benchmarking. This sets the standard for how to evaluate and which measures to compare.

### 3.4.1 Validation of segmentation



**Figure 3.15:** Flowchart of the segmentation validation

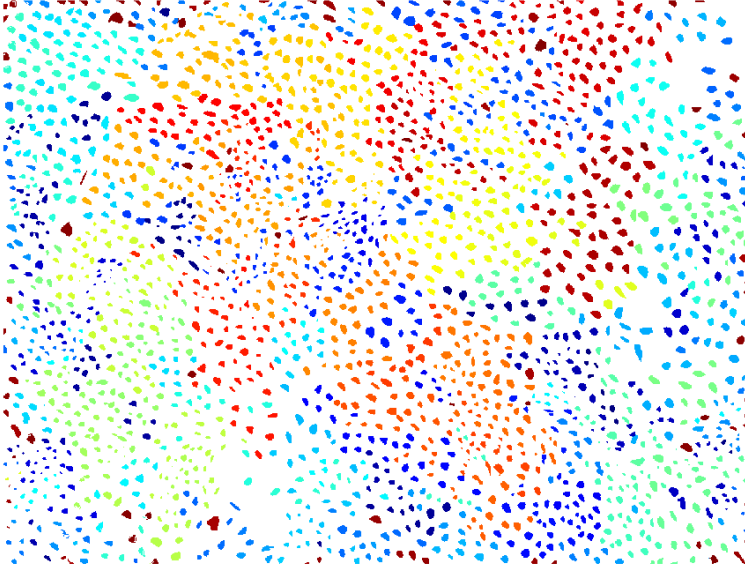
The result of the segmentation is a cell mask containing all segmented cells, each with a unique number as seen in Figure 3.16.

The validation of the segmentation amount to ensuring a unique correspondence between the segmented cells and the true cells. It is not likely a 100% correspondence, so a a measure of sufficient overlap between a segmented cell and the corresponding true cell is needed. The cell is categorized as correctly detected if it overlaps one and only one cell in the reference cell mask and the overlap in both directions exceeds a threshold of 30%. The cells are classified as TP, TN, FP, FN dependent on their overlap.

where:

- TP is True Positive - Correctly detected cells
- TN is True Negative - No cell exist and no cell is detected - not tested for!
- FP is False Positive - Detected cell that does not exist in the reference
- FN is False Negative - Cell that exist in the reference but is not detected

The True Negative is not tested for, but it can be calculated from the other entities After implementing the cell validation some tests where performed on a



**Figure 3.16:** Fully segmented image, timeframe 200

constructed test set to verify the functionality. Some of these the tests can be seen in Figures 4.12,4.13 and 4.14. To be sure that the needed performance was achieved no matter which order the individual tests was performed in, the tests has been run where the true and detected data was swapped and also where the segments where numbered in different order. The results where the same for each pair of test and detection data. The tests covered the 4 possible scenarios:

- One true cell is segmented as one.
- Two true cells are segmented as one.
- One true cell is segmented as two.
- Overlap less than 30% - not detected.

The algorithm test not only for if one true cell is segmented as two, but for if one true cell is segmented as more than one and the same in the opposite direction. The detection rate in percent is also calculated from:

$$detectionrate = 100 \cdot \frac{Correctly\ detected\ cells}{Number\ of\ true\ cells} \quad (3.13)$$

See section 4.1.5 for examples and tests.

### 3.4.2 Validation of tracking

Validation of the tracking begin by assuming the paths are only complete and valid if they start from a mitosis and also end with a mitosis. If this is not the case the path is only a fragment of a correct path and can not pass the validation as it is. A path that start in the first frame or end in the last frame are the exception from this and are also included. It is furthermore assumed that the paths should be identical to the paths from the true solution. In order to achieve this, each and every cell in a path should correspond to a unique cell in the path of the true solution. If only one cell fail, the hole of the path fails, and is not registered as correct.

The complete paths are picked out from the Save Matrix and one by one checked for cell correspondence. In the true solution the cells are numbered according to which path they belong to. The correspondence in the first cell is saved and it is assumed that the rest of the path in the true solution has the same number. This is verified one cell at a time from the cell masks. The overlap between the two cells is calculated and the median of the overlap is used to determine the correspondence, in case of the cells overlap more than one other cell. If all cells in a path pass the validation, the path is marked as correct.

## CHAPTER 4

# Test

---

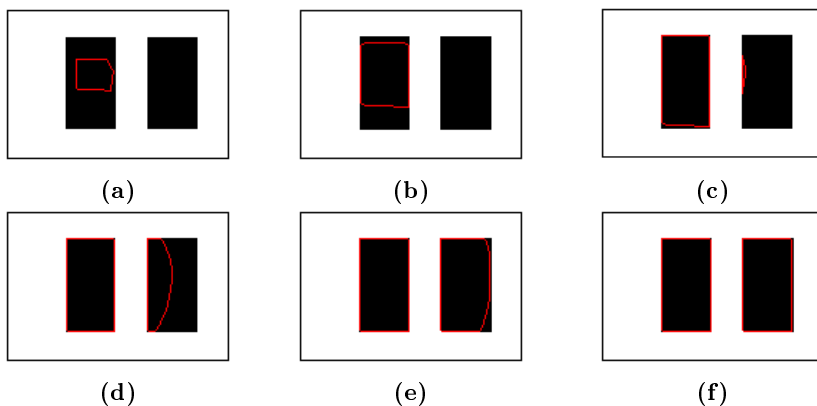
Testing is an important part of software development. Testing is a means of verifying if the software fulfill the specified requirements. There is always a risk that the software might be erroneous, but testing reduce this risk. The areas tested are chosen to verify only the reliability and correctness of the code. Tests can only show presence of errors. Manual debugging is necessary to determine the cause of error after it has been detected. Testing should be performed as soon as possible since the cost will be lower at an early stage of developing the software both in term of price but also time needed to correct for the found errors. The following tests performed in this chapter only show a verification of the specified functionality. Good coding practice has been observed during implementation, but the focus has been kept on functionality and not so much on fast execution time.

## 4.1 Segmentation

### 4.1.1 Test 1

Test 1 is performed to see how the algorithm performs with respect to the initialization and the segmentation it self. A test image with two black square objects on a white background is constructed and the algorithm is applied to this image. It is assumed that cells in the RefdataA dataset is overlapping from one time frame to the next. This is not the only case that is tested for but also:

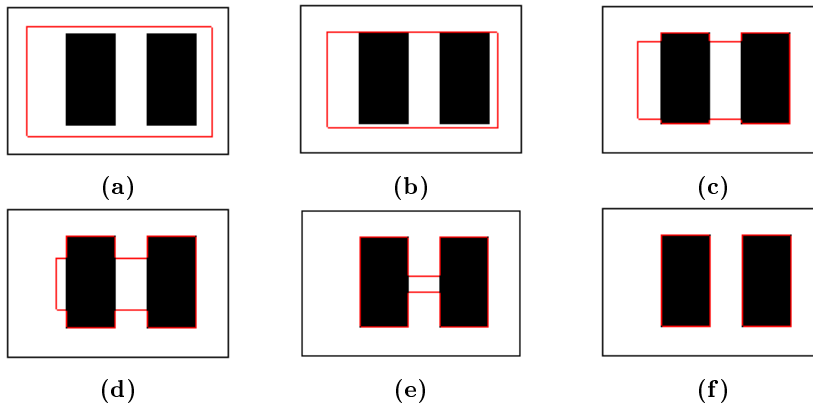
- Initial contour inside the objects
- Initial contour around the objects
- Initial contour outside the objects
- No segments present
- Initial contour overlapping the objects



**Figure 4.1:** Initial contour inside object

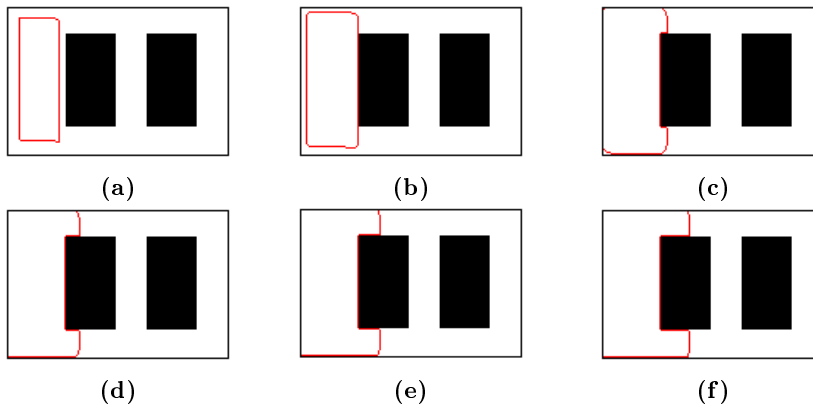
From Figure 4.1 it is seen that the contour is expanding to segment not only the left object but over time the external force make it move beyond the white background, till it finally segments both objects. Individual contours are colored with different colors, so since the two contours are the same color they belong to the same contour. This is a good feature for an initial segmentation where all objects should be found in an image and also for a global search for new appearing segments.





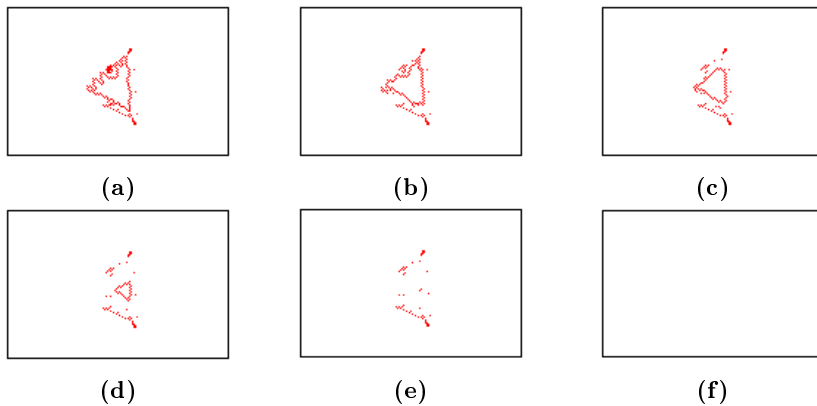
**Figure 4.2:** Initial contour around segments

In Figure 4.2 the initial contour is around the segments and it is seen that the algorithm adapts and segment both objects.



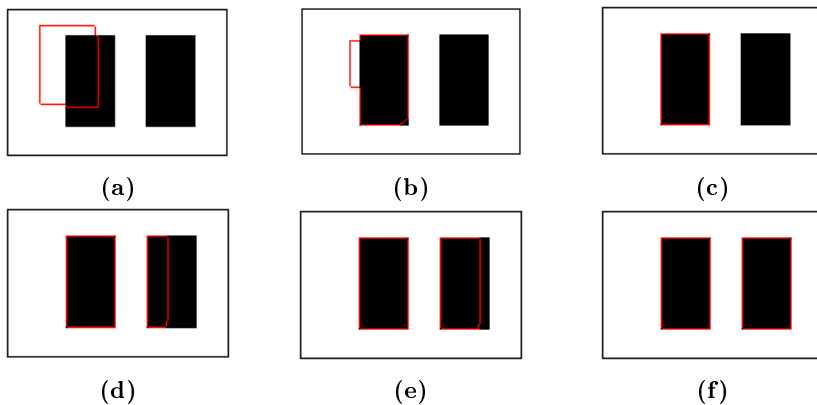
**Figure 4.3:** Initial contour outside segments

Figure 4.3 show that if the initial contour is not overlapping with the objects at all, but there is a significant difference in the mean values of the foreground and back ground, the area inside the contour is taken to be the foreground and segmented accordingly. This situation will in practice not happen.



**Figure 4.4:** No segment present

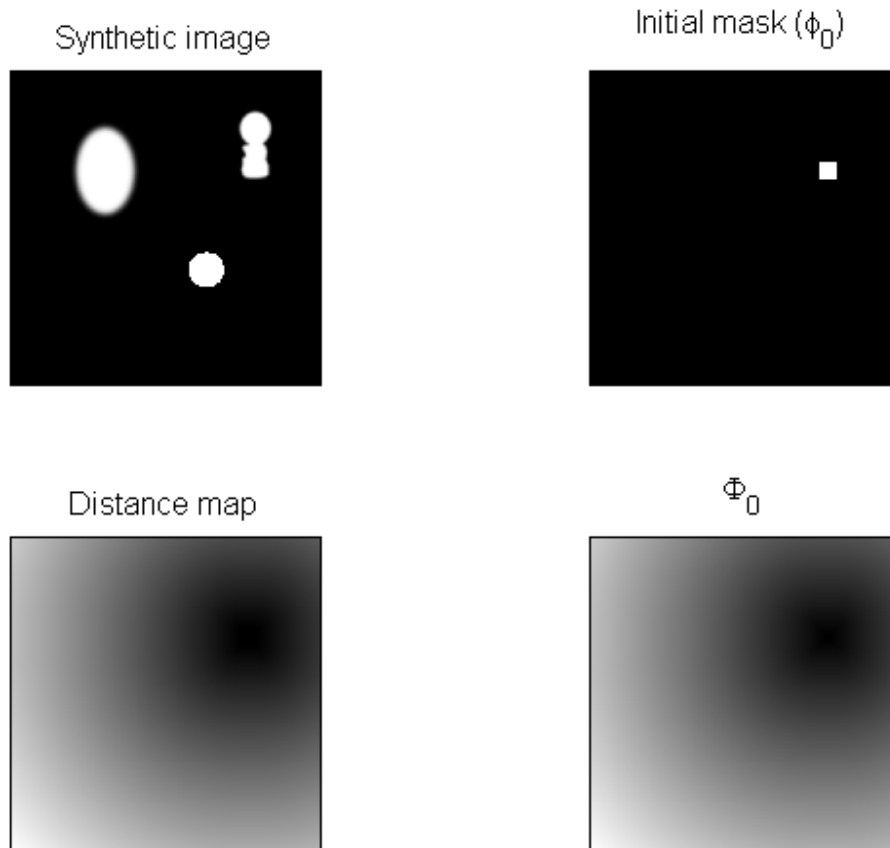
It is seen from Figure 4.4 that in the case where there is no significant difference in the mean value of the outside and the inside of the contour, only the internal force will act upon the contour and this will make it evolve in the negative normal direction, causing the contour to shrink and ultimately vanish.



**Figure 4.5:** Initial contour overlapping segment

The final images of test 1 seen in Figure 4.5 Show the normal case where the initial contour is overlapping with the object and this works fine, the overlapped object is segmented but also the other object which should be segmented by another individual contour. The algorithm needs to be refined to accommodate this.

## 4.1.2 Test 2

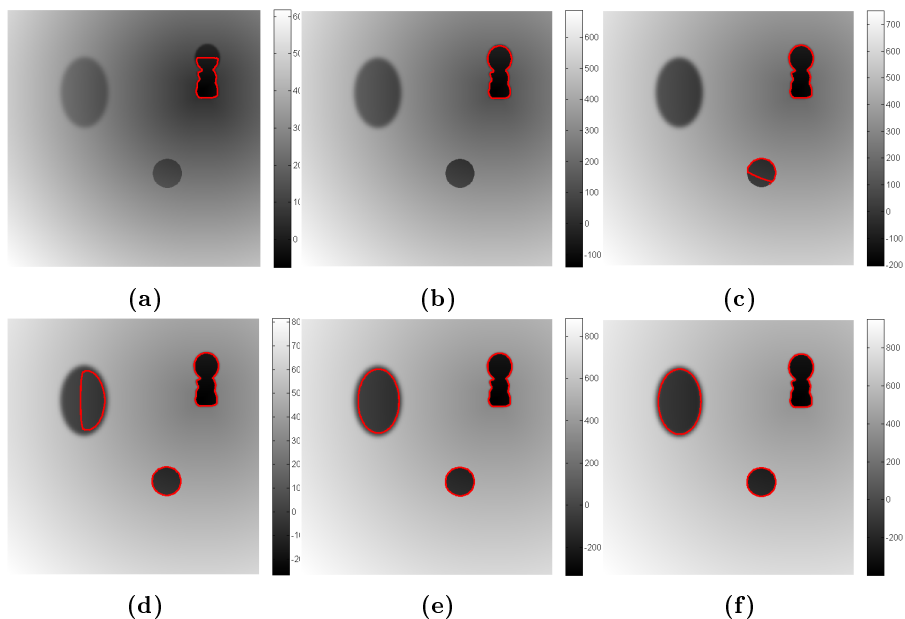


**Figure 4.6:** Initial states of test 2

Figure 4.6 top left, show the test image that was constructed for this test. There are three objects an oval with gaussian blurred edges to test if the model work without sharp edges. There is a circle with sharp edges and the third object has an outline that is not smooth.

The top right image show the initial mask, that will be evolved. The two bottom images are the same, they show that the initial state of the function  $\phi$  is the distance function it self.

Figure 4.7 show the evolution of the function  $\phi$  and the image force that it is



**Figure 4.7:**  $\phi$  and force evolved over time

updated with each iteration. It is seen that the force get stronger and stronger over time. Each pixel in all the objects, in the test image have the same intensity and hence the same force is added to these pixels. The segmentation is a result of the distance function and the force added each iteration, so the difference in when the different objects are segmented is due to the distance function.

The segmentation work well in the oval shaped object with blurred edges, the circle and also in the last object with a non smooth outline.

### 4.1.3 Test 3

The algorithm has been refined so that it can evolve several contours independently. Now it should be tested for if this is the case. This is also performed on the same test image. Now the contour is assigned a different color for each separate segment. Now the initial mask that is evolved is placed in the oval shaped object.

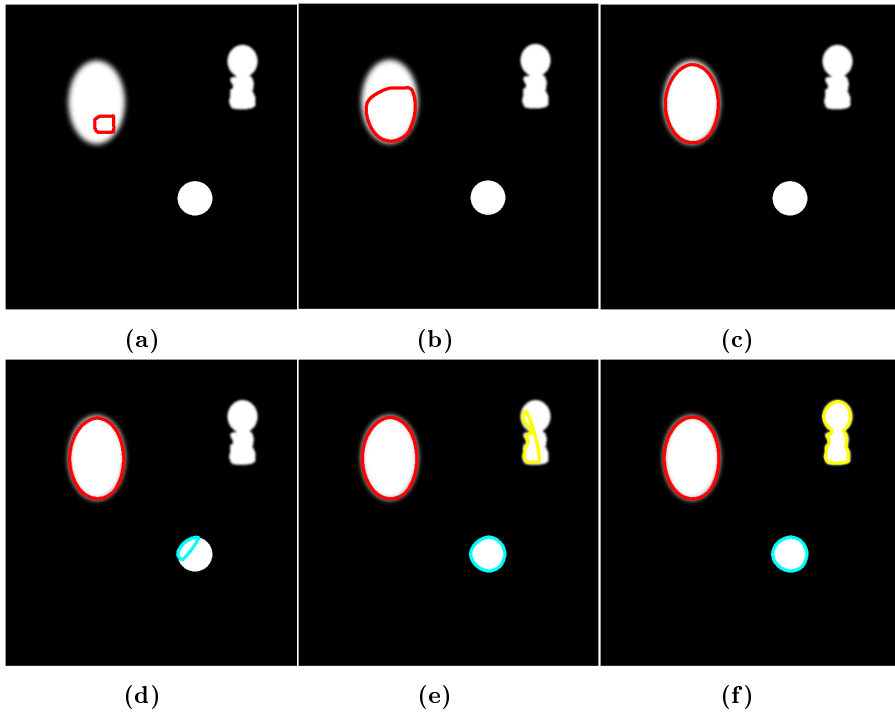


Figure 4.8: Initial contour overlapping segment

#### 4.1.4 Test 4

The fact that two cells can not merge, but need to be kept separate if they occasionally touch should be tested for.

The two test images used are again black and white images, one with a large square and one where the large square is divided into two smaller squares as seen in Figure 4.9

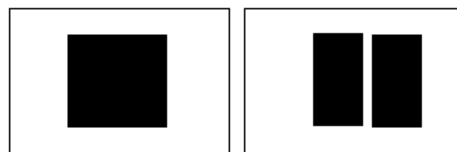
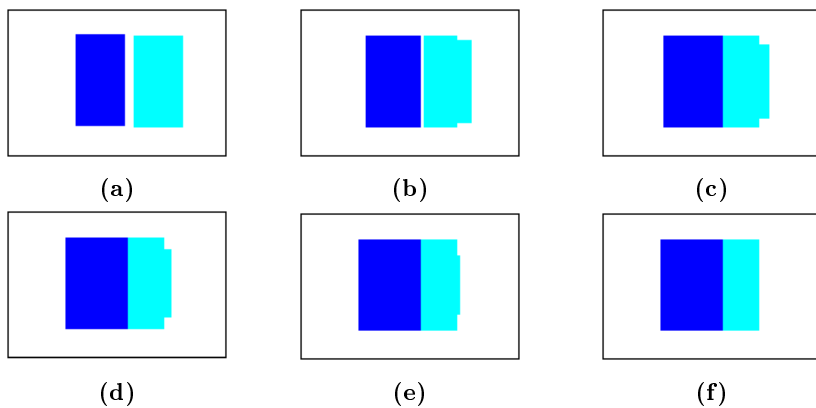


Figure 4.9: Flowchart of the segmentation validation

At first the contour is initialized with a contour around the two squares and

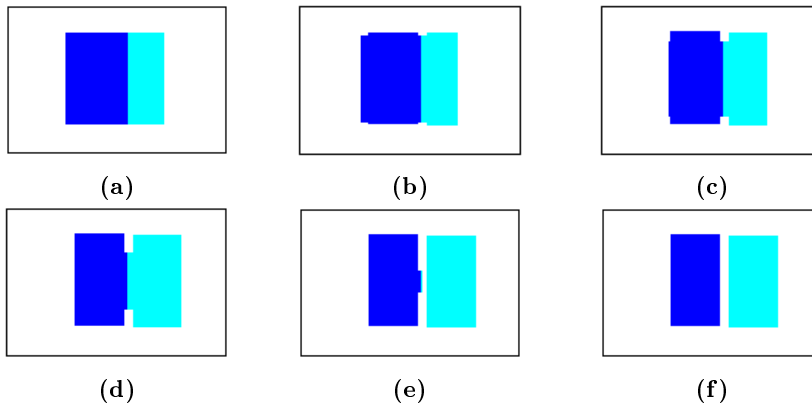
the active contour model is applied to the image with the big square to see if the contours merge. Instead of displaying the outline of the contour, the whole segmented area is displayed. This is less demanding computations since the Heaviside function is used and the Heaviside function is already calculated in the active contour model.



**Figure 4.10:** Test for merge

It is seen from figure 4.10 that even though the two contours share the same area they do not merge.

Now the next test is to see if the contours can split again after they have touched. For this test the previous segmentation is used for the initializer and the active contour model is applied to test image with two squares.



**Figure 4.11:** test for split

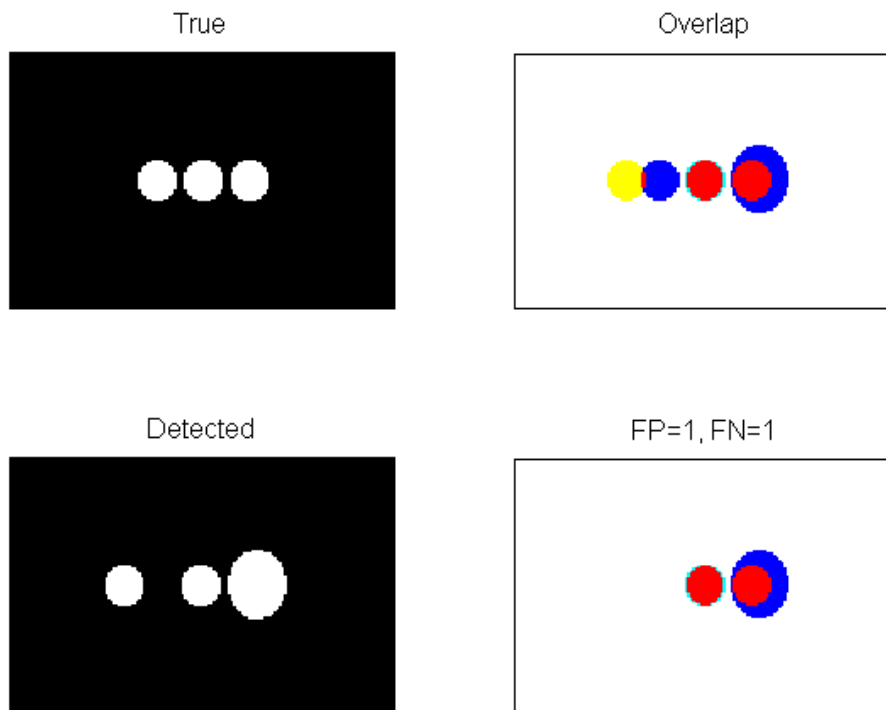
As seen in Figure 4.11 the contours split nicely again. This is a simple case and it can be expected that if the two objects are moved or rotated while touching, the result will not be so good.

Now the basic required properties have been tested and verified, now the active contour model can be applied to the real dataset.

### 4.1.5 Validation of segmentation

The following tests each show four images. The top left image show the reference data. The bottom left show the detected cells. The top right is the overlap between the reference and detected cells. The bottom right is overlap between the true reference with deletion and the validated correct detected cells with the number of errors in the title. The last image show only the correct detected cells all other have been deleted.

#### 4.1.6 Test 5

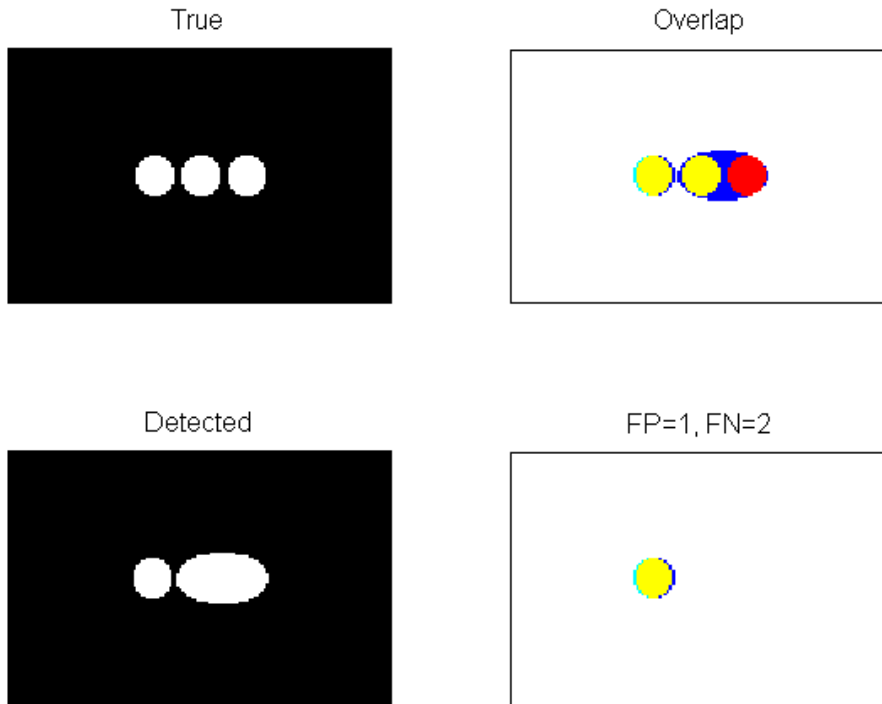


**Figure 4.12:** Segmentation Validation Test 5.

Figure 4.12 show a test for overlap less than 30 %. The left detected segment is not overlapping the true reference segment and cause a FP error. The true segment is therefore not segmented and cause a FN error. The other segments overlap more than 30% in both directions and are accepted as correctly segmented and cause a TP. The acceptance rate is 66.66%



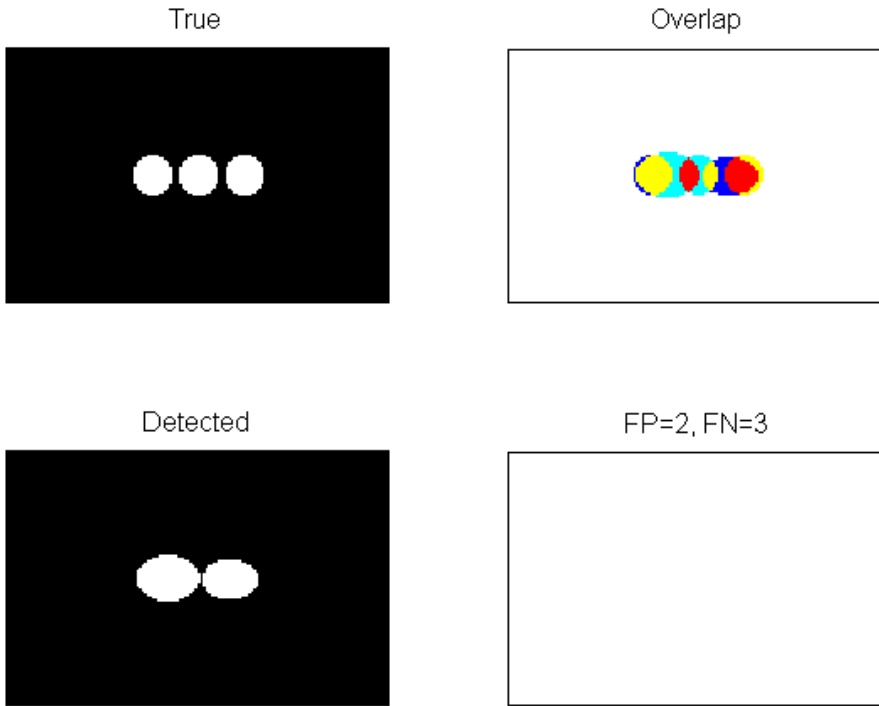
## 4.1.7 Test 6



**Figure 4.13:** Segmentation Validation Test 6.

Figure 4.13 show a test for when two true cells are segmented as as one. The right detected cell is overlapping two cells and cause an FP error and is deleted. The two right true cells are then not detected and cause two FN errors and are deleted. The remaining cell is correctly detected and cause a TP. The acceptancerate is 33,33%.

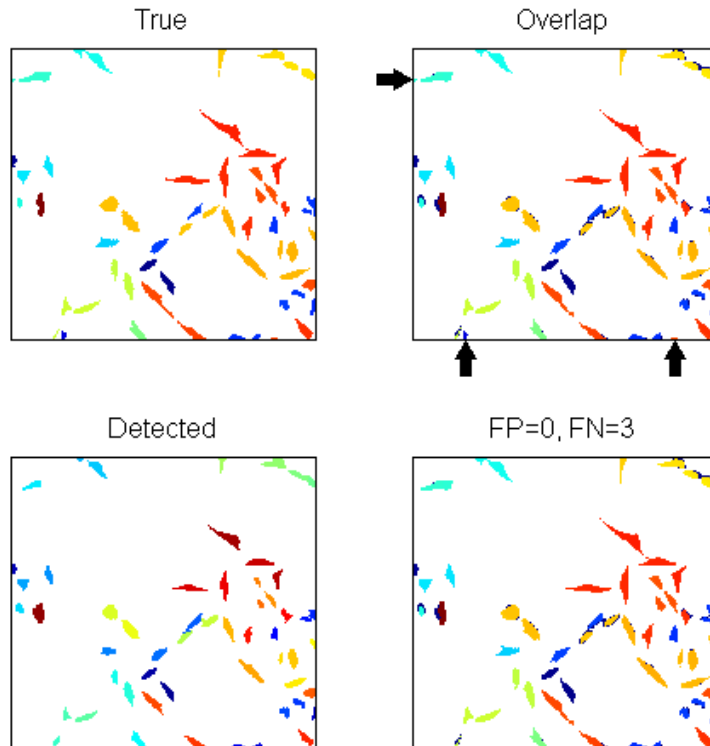
## 4.1.8 Test 7



**Figure 4.14:** Segmentation Validation Test 7.

Figure 4.14 show a test for both two cells detected as one and one cell detected as two. The middle true cell is overlapped by two and cause one FN error. The two detected cells both overlap two and cause two FP errors. The three of them are deleted and the two remaining true cells are not segmented so they cause two FN errors. The detectionrate is 0% hence there are no TP.

## 4.1.9 Test 8



**Figure 4.15:** Segmentation Validation Test 8.

Figure 4.15 show a test on a subset of the RefdataA dataset. There are three segments that are deleted because they are on the image border and smaller than 50 pixels. They are of course registered as FN errors. The cells in question are marked with black arrows. The test was run several times on different time frames and in general most of the cells are correctly segmented. I did not find any errors from manual inspections. This is an indicate that the segmentation is quit good, but it makes it difficult to test the validation with the RefdataA data set. It was a better choice to use the artificial dataset from the previous to test the validation.

## 4.2 Results

### 4.2.1 segmentation

The results from the validation of the segmentation are summed up in a calculation of the detection rate:

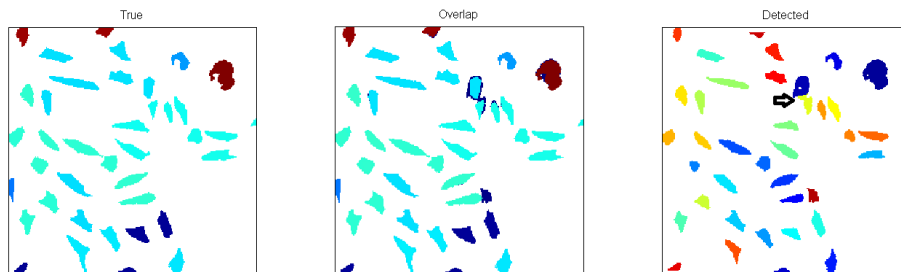
$$detectionrate = 100 \cdot \frac{\text{Correctly detected cells}}{\text{Number of true cells}} \quad (4.1)$$

The detection rate is the accumulated detection over all 209 image frames. The resulting detectionrate is :

Accumulated detection rate = 87,76%.

This is quite low compared to the benchmark which is above 95%

One reason this is low is the validation criterion which say that a cell can only overlap one and only one cell. This is not possible when cells are allowed to touch without merging the overlap will sometimes be violated and count as an error. And also when they split again after touching some times a small fraction of one segment is still attached to the other segment, which again violate the validation criterion.



**Figure 4.16:** Overlap error

As seen in figure 4.16 at the arrow, two cells are touching with only a small overlap. This is enough for the validation to cause both of the cells to be erroneous and the two remaining true cells are hereby not segmented and they cause another two errors. One small overlap cause four errors. This is not good for the detection rate.

Allowing cells to touch was one of the improvements compared to the benchmark, but the validation criteria are not favoring this.

### 4.2.2 Tracking

The result of the tracking is also summed up in a detection rate:

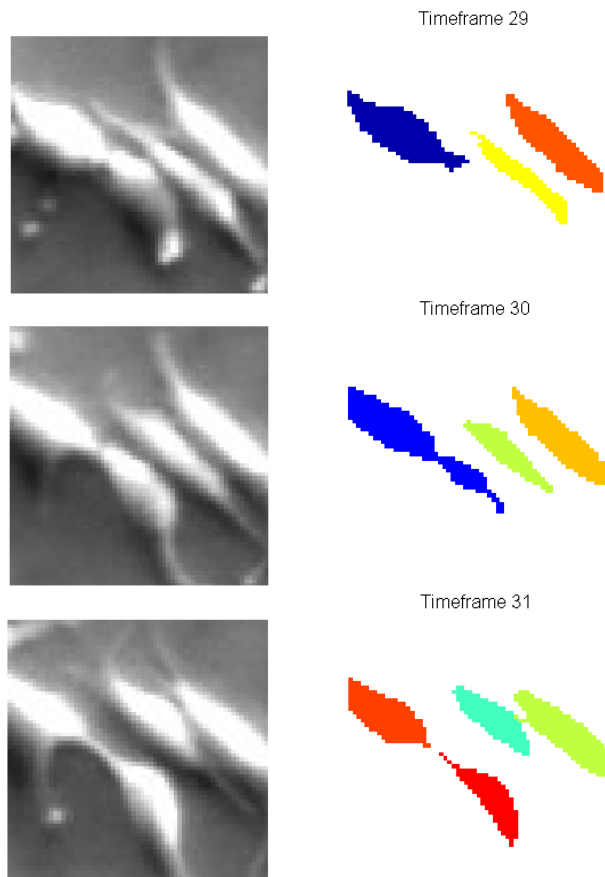
$$detectionrate = 100 \cdot \frac{\textit{Correct complete paths}}{\textit{complete paths}} \quad (4.2)$$

The calculate detection rate was at first 0%, this after thorough investigation turned out to be caused by a missing correspondence in the time of detection of mitoses. The true data is produced by a threshold by otsu's method and afterwards the cells are eroded. This sometimes cause otherwise connected cells to split into two separate cells. An example can be seen in the following figures.



**Figure 4.17:** Validation problem, timeframe 29

Figure 4.17 show the two segments at the arrow are completely separated at time frame 29. Figure 4.18 is a zoom in on these cells and their progression over time, and the segmentation. The question is when to register the mitosis ?.



**Figure 4.18:** Validation problem

Figure 4.18 show the segmentation separates the two cells and register the mitosis at timeframe 31, but the cells are still connected by an axion. Detecting the mitoses is a problem.

To see if this would improve on the detectionrate a simple solution was to not validate the last two cells in a path. This immediately gave the result:

Detectionrate = 39.62%

This indicate that the error was correctly identified, but less than 40% is nor a good result. The remaining correct paths where investigated and the common error was the path was followed nicely to a certain point and then it suddenly changed to another path and continued from here on. This indicate a problem

---

in the numbering of the cells. The implementation was thoroughly checked but no software bugs found. It turned out that objects with an area less than 50 pixels are deleted since they are assumed to be debris. Occasionally a cell shrink to a size below the threshold, and is deleted, but not registered as ending. This cause the cells to be renumbered wrongly and all the paths with a higher number going through this time frame are corrupted.

A solution to this problem could be to handle the deletion of the debris more intelligently or to number cells differently.





# Conclusion

---

The purpose of this thesis was to construct a pipeline for segmentation and tracking of cells in microscopic images. This part was completed, but it is obvious from the results, that it does not perform to the required level of performance. This is due to several implications in the approach and also in the complexity of the dataset.

The segmentation performs better than the benchmark, and this was expected, since the segmentation in the benchmark was done by a simple threshold performed by Otsu's method. The active contour model have the advantage of physically being able to link two frames together to perform the segmentation, which improve the result. This also provide a smooth way of tracking the cells, which has an elegant continuous mathematical formulation that can be proved.

One approach that proved not to be good was the unconditional deletion of debris. A size restriction by a threshold of 50 pixels was used to determine if an object was a small cell fragment or debris. This was how the debris was handled in the benchmark. It turned out that in some cases a cell could be deleted unintentionally if it moves upwards and go out of focus or if the light intensity is low. this combined with the strategy of renumbering the cells in each frame led to the wrong renumbering of some of the cells. The implications of this where wide spread, and the cost a large reduction in the measured correct classified paths.

Tracking of the cells are naturally performed by the active contour model, since it evolves from a cell in one frame to a cell in the the next frame. This provide the linkage between the cells in each frame. The results could not be measured exactly, but manual inspections of the erroneous paths show that most often the correct cells are linked but the numbering is wrong. This again imply that the renumbering strategy was a bad approach, but the active contour model live up to the standard.

Major forces of the active contour model is it's ability to register changes in topology. Mitosis, appearance and disappearance of cells are easily captured by the model and are fairly easy to detect. However validating the mitoses proved to be more difficult than first assumed. The cells are partially connected in the stages after mitosis and deciding when to separate the new cells was an issue.

The validation procedure was implemented and it serve as a measure of the quality of the segmentation and tracking. Unfortunately the results are not true, but affected by the unintentional renumbering of cells. The segmentation was tested for position and overlap compared to the benchmark but not for the numbering. The tracking is only tested in the validation procedure so the errors was unnoticed till the end of the project and only captured in the validation procedure

The true data supplied has been produced by thresholding and morphological operators that some time separate connected cells . The active contour model segments the data differently, and does not use the morphological operators, so the occurrence of a mitosis is not registered in the same time frame. This was an issue that had to be addressed and a workaround was introduced.

A chain is only as strong as its weakest link.

Since the renumbering failed it was not possible to validate the solution properly with a valid result.

The analysis done on the errors do not indicate any problems regarding the use of the active contour model, so it is recommended that the model is developed further. The active contour model show great potential in the tracking if the numbering of the cells can be corrected or the numbering procedure changed. The mitosis detection is also quite good, but again this could not be quantitatively determined by the validation procedure because of the errors.

## CHAPTER 6

# Future work

---

The future work is first and foremost to handle the errors pointed out in the previous chapters.

A new validation procedure should be developed, that favors the features of the active contour model.

When the pipeline is fully functional, the parameters should be tweaked to an optimum. This could be obtained by running the pipeline for different values of parameters to get an estimate to where to start. A real optimization procedure for adjusting the parameters could also be developed. This will enhance the performance.

Some of the procedures from the segmentation association techniques could link the paths fragments, the fragments could be combined into complete paths, hereby improving the tracking.



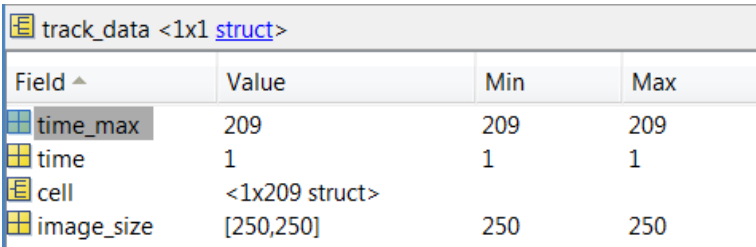
# Appendix1

---

## 7.1 Structures

### 7.1.1 Track\_data

The segmentation is completed and the resulting data is saved in the struct `track_data`. The struct provides a logical and easy access to the data for performing the tracking.



Field ^	Value	Min	Max
time_max	209	209	209
time	1	1	1
cell	<1x209 struct>		
image_size	[250,250]	250	250

**Figure 7.1:** Track\_data struct

The first "layer" of the `Track_data` struct also contain information on image size and the first and last timeframe. The struct cell contain the segmented

data for each timeframe as seen in Figure 7.2

Field ^	Value	Min	Max
centroid	<9x2 double>	5.4023	236.2275
cells	<1x9 cell>		
neighbour	[ ]		
successor	[1,2,3,4,5,6,7,8,9]	1	9
predecessor	[0,0,0,0,0,0,0,0]	0	0
number	9	9	9
status	[1,1,1,1,1,1,1,1,256]	1	256

**Figure 7.2:** Track\_data.cell

The track\_data.cell structs contain the actual cell masks, the centroid of each cell, no. of cells and their status and most important for the tracking their successor, which is linking the cells between the current timeframe and the next timeframe.

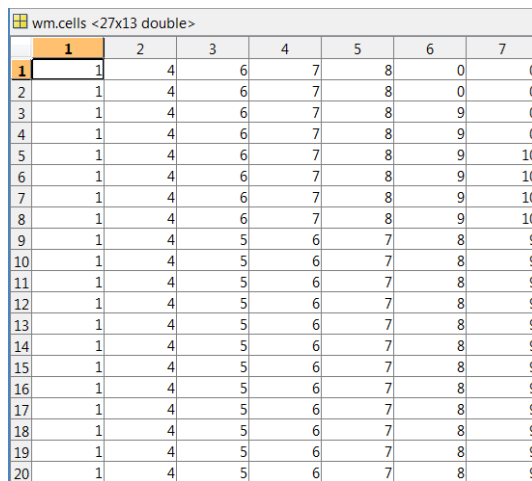
#### 7.1.1.1 status

The status is determined during the tracking and has to be accumulated and updated on the path level. The statuses are as follows:

int value	status-flag
0	empty path, not tracked
1	path
2	mitosis
4	border begin
8	border end
16	NOT USED
32	corrected
64	ending
128	beginning
256	lost begin
512	lost end
1024	coupled

### 7.1.2 Work matrix

During the segmentation the cells in each timeframe was renumbered and this has to be taken into account in the tracking. The work matrix is therefore created iteratively one timeframe at a time. When ever one complete track is present it is saved to the save matrix and deleted from the work matrix to accommodate the renumbering. a matrix is used since it fits the format of the data with time as rows and cell numbers (tracks) as columns. The iterative approach take up less memory and enable that a full column containing a path can be deleted.



	1	2	3	4	5	6	7
1	1	4	6	7	8	0	0
2	1	4	6	7	8	0	0
3	1	4	6	7	8	9	0
4	1	4	6	7	8	9	0
5	1	4	6	7	8	9	10
6	1	4	6	7	8	9	10
7	1	4	6	7	8	9	10
8	1	4	6	7	8	9	10
9	1	4	5	6	7	8	9
10	1	4	5	6	7	8	9
11	1	4	5	6	7	8	9
12	1	4	5	6	7	8	9
13	1	4	5	6	7	8	9
14	1	4	5	6	7	8	9
15	1	4	5	6	7	8	9
16	1	4	5	6	7	8	9
17	1	4	5	6	7	8	9
18	1	4	5	6	7	8	9
19	1	4	5	6	7	8	9
20	1	4	5	6	7	8	9

**Figure 7.3:** True data, Segmentation, overlap and validated with overlap

From Figure 7.3 it can be seen that the paths represented by cell 2 and 3 are deleted and also the path from row 1 to row 8 in the third column is deleted and saved in the save matrix. This path is the first path in the save matrix in Figure 7.4

### 7.1.3 Save matrix

The save matrix is actually a cell array. The cell array allow for each cell to be of different size and type. It holds the time\_begin, time\_end, cell id's and children of the paths.

sm <482x4 cell>				
	1	2	3	4
1	1	8	[5;5;5;5;...]	0
2	9	9	10	0
3	10	10	10	0
4	11	11	10	0
5	14	15	[11;11]	0
6	14	15	[11;11]	0
7	15	18	[12;0;0;11]	0
8	1	20	<20x1 do...>	[10,11]
9	14	24	<11x1 do...>	0
10	1	25	<25x1 do...>	[11,12]
11	21	26	[12;12;12;...]	0
12	26	26	11	0
13	26	27	[14;13]	0
14	26	28	[13;11;12]	0
15	1	30	<30x1 do...>	[13,14]
16	26	32	[11;10;10;...]	[16,17]
17	33	34	[17;17]	0
18	30	39	[12;11;11;...]	0
19	1	40	<40x1 do...>	[15,16]
20	31	40	[15;15;14;...]	[17,18]

Figure 7.4: The first 20 rows of the save matrix



# Bibliography

---

- [CV01] Tony F. Chan and Luminita A. Vese. Active contours without edges. February 2001.
- [KMS02] K.F.Riley, M.P.Hobson, and S.J.Bence. *Mathematical methods for physics and engineering*. Cambridge, 2002.
- [MDSvC09] E Meijeringa, O Dzyubachyka, I Smala, and W A. van Cappellen. Tracking in cell and developmental biology. August 2009.
- [MS89] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. 1989.
- [OF03] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*. Springer, 2003.
- [PRR10] D Padfield, J Rittscher, and B Roysam. Coupled minimum-cost flow cell tracking for high-throughput quantitative analysis. August 2010.
- [Ran05] Rangaraj M. Rangayyan. *Biomedical Image Analysis*. CRC PRESS, 2005.
- [RBM<sup>+</sup>11] D.H. Rapoport, T. Becker, A.M. Mamlouk, S.Schicktanz, and C. Kruse. A novel validation algorithm for automated cell tracking and the extraction of biologically meaningful parameters. November 2011.