IMAGE INPAINTING WITH THE COMPLEX GINZBURG-LANDAU EQUATION

by

Sonoko Nakano
BS., Simon Fraser University, 2008

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the Department
of
Mathematics

© Sonoko Nakano 2010
SIMON FRASER UNIVERSITY
Summer 2010

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without the permission of the author.

APPROVAL

Name:	Sonoko Nakano
Degree:	Master of Science
Title of Thesis:	Image Inpainting with the Complex Ginzburg-Landau Equation

Examining Committee: Dr. Paul Tupper
Chair

____

Dr. Manfred Trummer,
Senior Supervisor

____

Dr. Steve Ruuth,
Supervisor

____

Dr. JF Williams,
Internal/External Examiner

Date Approved: August 4, 2010

SFU SIMON FRASER UNIVERSITY
LIBRARY

Declaration of Partial Copyright Licence

The author, whose copyright is declared on the title page of this work, has granted to Simon Fraser University the right to lend this thesis, project or extended essay to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users.

The author has further granted permission to Simon Fraser University to keep or make a digital copy for use in its circulating collection (currently available to the public at the "Institutional Repository" link of the SFU Library website <www.lib.sfu.ca> at: <http://ir.lib.sfu.ca/handle/1892/112>) and, without changing the content, to translate the thesis/project or extended essays, if technically possible, to any medium or format for the purpose of preservation of the digital work.

The author has further agreed that permission for multiple copying of this work for scholarly purposes may be granted by either the author or the Dean of Graduate Studies.

It is understood that copying or publication of this work for financial gain shall not be allowed without the author's written permission.

Permission for public performance, or limited permission for private scholarly use, of any multimedia materials forming part of this work, may have been granted by the author. This information may be found on the separately catalogued multimedia material and in the signed Partial Copyright Licence.

While licensing SFU to permit the above uses, the author retains copyright in the thesis, project or extended essays, including the right to change the work for subsequent purposes, including editing and publishing the work in whole or in part, and licensing other parties, as the author may desire.

The original Partial Copyright Licence attesting to these terms, and signed by this author, may be found in the original bound copy of this work, retained in the Simon Fraser University Archive.

Simon Fraser University Library
Burnaby, BC, Canada

Last revision: Spring 09

Abstract

Image inpainting is the process of reconstructing lost or deteriorated parts of an image or video. The topic of this thesis is a mathematical technique for inpainting based on the Ginzburg-Landau equation (GL). This approach was first presented by Grossauer and Scherzer. The Ginzburg-Landau equation is a classical equation in the field of electromagnetism, introduced by V.L. Ginzburg and L.D. Landau in 1950. At first the Ginzburg-Landau energy is discussed, and then it is shown how to obtain the GL equation by minimizing this energy. This is followed by a brief discussion of the mathematical properties of the GL equation. It is then shown how to apply the real and complex GL equation to inpainting. Our approach is essentially to compute the steady state of an evolution equation. Finite differences are employed to discretize the equation, and the stability and consistency of an explicit and an implicit scheme is analyzed. Finally, some examples are presented, and the performance of our inpainting method is discussed. A comparison between the GL-based method and the one introduced by Bertalmio, Sapiro, Caselles and Ballester concludes the presentation and indicates the GL approach is superior.

Acknowledgments

I would like to give my sincere thanks to my supervisor Dr. Manfred Trummer for his suggestion of the topic, for his patience, guidance and encouragement in the writing of this thesis, and for the effort he made in reading and correcting my work. Without his support, this thesis could not have been completed.

Special thanks to all members of my examining committee, Dr. Paul Tupper, Dr. Steve Ruuth and Dr. JF Williams, for their invaluable comments and suggestions.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Image inpainting is the process of restoring lost or damaged parts of a picture, painting or frame in a video. It has traditionally been performed by artists. Digital image inpainting, first introduced in the paper of Bertalmio, Sapiro, Caselles and Ballester (SIGGRAPH 2000) ([1]), is done automatically by computer software.

The common use of digital inpainting is to restore missing data in the image, but inpainting techniques have broad applications from restoring deteriorated paintings and photographs to removing or replacing selected objects in an image. There are many different approaches proposed by different researchers.

This thesis describes inpainting using the Ginzburg-Landau equation based on the paper by Grossauer and Scherzer [20]. Other methods are:

1. An inpainting algorithm based on level lines (proposed by Masnou and Morel) [25, 26]

2. A variational method (proposed by Ballester et al.) [9]
   The algorithm solves a system of PDEs by using level sets of the image intensity function.

3. An algorithm which is like a professional restorator's work (proposed by Bertalmio et al.) [11] (This method is described in Chapter 5.)
   Let the region to be restored be Ω. The information surrounding Ω is prolonged into Ω by extending the contour lines at the boundary of Ω . The algorithm is combined with diffusion process to ensure a correct evolution of the direction field.

4. TV-inpainting (proposed by Chan and Shen) [14, 15]

The TV-inpainting model is closely connected to the total variation (TV) denoising model.

5. A method of a fourth order PDE (proposed by Esedoglu and Shen) [16]
A fourth order PDE was derived from the combination of the Euler elastica functional and the Mumford-Shah functional.

6. An inpainting algorithm which is optimized for speed (proposed by Oliveira et al.) [27]

1.1 Ginzburg-Landau Energy

The Ginzburg-Landau equation is a classic equation arising in the physics of electric magnetism. According to the papers [20, 19] and a book [29], it was introduced in [18] by V.L. Ginzburg and L.D. Landau in 1950. The original equation was for the Ψ-function for the 'superconducting electrons' [18], and the equation which was introduced in [18] is rather complicated.

In [22], the original Ginzburg-Landau energy with the magnetic field is defined as

$$\begin{aligned} E(\psi, A) &= \int_\Omega F_n + \alpha|\psi|^2 + \frac{\beta}{2}|\psi|^4 \\ &\quad + \frac{1}{2m^*}|(-i\hbar\nabla - \frac{e^*A}{c})\psi|^2 \\ &\quad + \frac{|\mathbf{curl}A|^2}{8\pi}, \end{aligned}$$

where, ψ is the wave function for the particle of charge e^* and mass m^*, c is the light speed, $\hbar$ is the Planck constant, and A is the vector potential for the magnetic field h, i.e. $h = \mathbf{curl}A$.

Since the topic of this thesis is not superconductivity, we start with the model of the Ginzburg-Landau energy as introduced in [30].

1.2 Ginzburg-Landau Energy with Magnetic Field

In [30], the Ginzburg-Landau energy with magnetic filed is introduced as

$$G_\varepsilon(u,A) = \frac{1}{2}\int_\Omega |\nabla_A u|^2 + |h - h_{ex}|^2 + \frac{(1-|u|^2)^2}{2\varepsilon^2}. \tag{1.1}$$

- The unknown u is a complex valued function, and in [18], it is denoted as Ψ, which is a characteristic parameter. For temperature above the critical temperature, T_c, $\Psi = 0$ in the state of thermodynamic equilibrium, while for temperature below T_c, $\Psi \neq 0$.
  The function u is normalized to be $|u| \leq 1$, and where $|u| \simeq 1$ the material is in the superconducting phase, while where $|u| = 0$, it is in the normal phase.
- The unknown A is the electromagnetic vector-potential of the magnetic field, mapping from Ω to $\mathbb{R}^2$.
- The notation ∇_A denotes the covariant gradient $\nabla - iA$: $\nabla_A u = \nabla u - iAu$. Therefore, $\nabla_A u$ is a vector with complex components ([30]).
- The parameter $h_{ex} > 0$ represents the intensity of the applied field.
- The parameter $\varepsilon = \frac{1}{\kappa}$ is the inverse of the "Ginzburg-Landau parameter", κ, a dimensionless parameter in [18].

1.3 Ginzburg-Landau Energy without Magnetic Field

In [20], the Ginzburg-Landau energy functional is introduced as the functional which depends on the *order function u:* $\Omega \to \mathbb{C}$:

$$F(u, \nabla u) := \frac{1}{2}\int_\Omega \underbrace{|-i\nabla u|^2}_{kinetic\, term} + \underbrace{\alpha|u|^2 + \frac{\beta}{2}|u|^4}_{potential\, term} \tag{1.2}$$

where, $\alpha = -\frac{1}{\varepsilon^2}$ and $\beta = -\alpha$.

In [30] as well as in [13, 22, 29], the Ginzburg-Landau energy without magnetic field is introduced as

$$E_\varepsilon(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 + \frac{(1-|u|^2)^2}{2\varepsilon^2}. \tag{1.3}$$

This equation is obtained by setting the magnetic potential A and the intensity of the applied field h_{ex} in $G_\varepsilon(u{,}A)$ equal to zero in (1.1). If we expand the second term in (1.3) and let $\alpha = -\frac{1}{\varepsilon^2}$ and $\beta = -\alpha$, we get,

$$E_\varepsilon(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 - \frac{\alpha}{2} + \alpha|u|^2 + \frac{\beta}{2}|u|^2. \tag{1.4}$$

F and E_ε in (1.2) and (1.3) respectively differ only by a constant $-\frac{\alpha}{2}$ in the energy term, therefore, they are equivalent: Minimizing either term gives the same minimizer.

1.4 Ginzburg-Landau Equation

The Ginzburg-Landau equation is a model equation for "Superconductivity" and "Phase transition", and it is derived from the Ginzburg-Landau Energy functional (1.2) or (1.3). (How to derive the Ginzburg-Landau equation from its energy functional is explained in the next chapter.)

The time dependent Ginzburg-Landau equation is given by:

$$\psi_t = \Delta\psi + \kappa^2(1 - |\psi|^2)\psi.$$

The stationary case is therefore

$$0 = \Delta\psi + \kappa^2(1 - |\psi|^2)\psi,$$

where κ is a positive constant, and ψ is complex valued function.

According to [13], in superconductors, $|\psi|^2$ is proportional to the density of superconducting electrons. i.e.

$$\begin{aligned} |\psi| \approx 1 &: \text{ Superconducting state,} \\ |\psi| \approx 0 &: \text{ Normal state.} \end{aligned}$$

In superfluids, $|\psi|^2$ is proportional to the density of the superfluid. Near the cores of the vortices, the superfluid density $|\psi|^2$ is almost zero; away from the cores, $|\psi|^2 \approx 1$.

Chapter 2

Minimizing the Ginzburg-Landau Energy

2.1 Euler Equation

As mentioned in Chapter 1, [20] says that Ginzburg and Landau derived the approximation for the corresponding energy functional, which depends on the *order function u:* $\Omega \to \mathbb{C}$:

$$F(u, \nabla u) := \frac{1}{2} \int_\Omega \underbrace{|-i\nabla u|^2}_{kinetic\, term} + \underbrace{\alpha |u|^2 + \frac{\beta}{2}|u|^4}_{potential\, term} \tag{1.2}$$

where, $\alpha = -\frac{1}{\varepsilon^2}$ and $\beta = -\alpha$.
The state of minimal energy satisfies the Euler equation of (1.2).

2.1.1 How to Calculate the Euler Equation

The Euler equation of an energy functional describes the stationary point of the energy functional. The factor $-i$ in the kinetic term is a holdover from quantum mechanics and is not essential ([20]), hence we omit $-i$.

We apply calculus of variation as in [17]. We construct $u_\epsilon(x) = u(x) + \epsilon\eta(x)$, where $\eta(x)$ is continuously differentiable and $\eta|_{\partial\Omega} = 0$. The quantity ϵ is a parameter which is independent of x. Substitution into (1.2) yields,

$$\begin{aligned} F &= F(u + \epsilon\eta, \nabla u + \epsilon\nabla\eta) \\ &= \frac{1}{2}\int_\Omega |\nabla u + \epsilon\nabla\eta|^2 + \alpha|u + \epsilon\eta|^2 + \frac{\beta}{2}|u + \epsilon\eta|^4. \end{aligned} \tag{2.1}$$

In order for F to be stationary at $\epsilon = 0$, it is necessary that

$$\frac{dF}{d\epsilon}|_{\epsilon=0} = 0. \tag{2.2}$$

Therefore, we differentiate (2.1) with respect to ϵ, and set $\epsilon = 0$.

$$\begin{aligned}
0 &= \frac{dF}{d\epsilon}|_{\epsilon=0} \\
&= \frac{1}{2}\frac{d}{d\epsilon}\Bigg[\int_\Omega (\nabla u \nabla u + 2\epsilon \nabla u \nabla \eta + \epsilon^2 \nabla \eta^2) + \alpha(u^2 + 2u\epsilon\eta + \epsilon^2\eta^2) \\
&\quad + \frac{\beta}{2}(u^4 + 4u^3\epsilon\eta + 6u^2\epsilon^2\eta^2 + 4u\epsilon^3\eta^3 + \epsilon^4\eta^4)\Bigg|_{\epsilon=0} \\
&= \frac{1}{2}\int_\Omega \Big(2\nabla u \nabla \eta + 2\alpha u \eta + \frac{\beta}{2} 4u^3\eta\Big) \\
&= \int_\Omega \Big(\nabla u \nabla \eta + \alpha u \eta + \beta u^3 \eta\Big) \qquad (2.3)
\end{aligned}$$

Using a single integration by parts for the first term, we let

$$\begin{aligned}
w &= \nabla u, & dv = \nabla \eta \\
dw &= \nabla\nabla u \;=\; \Delta u, & v = \eta.
\end{aligned}$$

Then, since $\eta|_{\partial\Omega} = 0$, we get,

$$\begin{aligned}
\int_\Omega \nabla u \nabla \eta &= \nabla u \eta\Big|_{\partial\Omega} - \int_\Omega \eta \Delta u \\
&= -\int_\Omega \eta \Delta u. \qquad (2.4)
\end{aligned}$$

Therefore, we plug (2.4) into (2.3) and we get

$$\begin{aligned}
0 &= \int_\Omega \Big(\nabla u \nabla \eta + \alpha u \eta + \beta u^3 \eta\Big) \\
&= \int_\Omega \Big(-\eta \Delta u + \alpha u \eta + \beta u^3 \eta\Big) \\
&= \int_\Omega \Big(-\Delta u + \alpha u + \beta u^3\Big)\eta. \qquad (2.5)
\end{aligned}$$

Because η is an arbitrary function, we must have:

$$-\Delta u + \alpha u + \beta u^3 = 0.$$

If we let $\alpha = -\frac{1}{\varepsilon^2}$ and $\beta = -\alpha$, then (2.5) becomes

$$\begin{aligned} \Delta u - \alpha u - \beta u^3 &= 0 \\ \Delta u + \frac{1}{\varepsilon^2} u - \frac{1}{\varepsilon^2} u^3 &= 0 \\ \Delta u + \frac{1}{\varepsilon^2}(1-u^2)u &= 0. \end{aligned}$$

When we use a complex-valued function for u, the energy term must be a real number, hence we have to use $|u|^2$ for u^2. Therefore, we get:

$$\Delta u + \frac{1}{\varepsilon^2}(1-|u|^2)u = 0. \tag{2.6}$$

Hence (2.6) is the Euler equation of $F(u, \nabla u)$ in (1.2). It is called the Ginzburg-Landau Equation.

2.2 Ginzburg-Landau Equation

We take a closer look at the Ginzburg-Landau equation (2.6).

2.2.1 $u(x)$ is a real-valued function

If u is a one-variable real-valued function, $u : \mathbb{R} \to \mathbb{R}$, the second term of (2.6) equals 0 when $u = \pm 1$ provided $\varepsilon \neq 0$. Since the solution to the first term is a harmonic function, and since we are looking at a one-variable function, $u_{xx} = 0$ implies u is a linear function. As the second term favours to ± 1, $u = \pm 1$ are the possible solutions.
One of the solutions for (2.6) is introduced in [20] as

$$u(x) = \frac{e^{\frac{\sqrt{2}}{\varepsilon}x} - 1}{e^{\frac{\sqrt{2}}{\varepsilon}x} + 1} = \tanh\Big(\frac{\sqrt{2}}{2\varepsilon}x\Big). \tag{2.7}$$

(Note that the hyperbolic tangent is $\tanh x = \frac{e^{2x}-1}{e^{2x}+1}$.) According to [2], the hyperbolic tangent is the solution of the nonlinear boundary value problem

$$\begin{aligned} &\frac{1}{2}u^{''} = u^3 - u \\ &u(0) = u^{'}(\infty) = 0. \end{aligned}$$

Here, we will show that

$$u(x) = \tanh\Big(\frac{\sqrt{2}}{2\varepsilon}x\Big) = \frac{e^{\frac{\sqrt{2}}{\varepsilon}x} - 1}{e^{\frac{\sqrt{2}}{\varepsilon}x} + 1}$$

is the solution of u of

$$u^{''} + \frac{1}{\varepsilon^2}(1 - u^2)u = 0.$$

If we rewrite the second order equation as a first order system,

$$\dot{u} = U(u, w), \qquad \dot{w} = W(u, w)$$

then the phase paths belong to the family described by the equation ([23])

$$\frac{dw}{du} = \frac{W(u, w)}{U(u, w)}.$$

In our case, we have the system $\dot{u} = w$, $\dot{w} = \frac{1}{\varepsilon^2}(u^3 - u)$, and we want to show that the solutions are given by $u = \pm \tanh\Big(\frac{\sqrt{2}}{2\varepsilon}x\Big)$.

The equilibrium points occurs when $\dot{w} = \frac{1}{\varepsilon^2}(u^3 - u) = 0$, i.e. a centre at $(0, 0)$ and two saddle points at $(\pm 1, 0)$. The phase path satisfies the separable differential equation

$$\frac{dw}{du} = \frac{\frac{1}{\varepsilon^2}(u^3 - u)}{w}.$$

When we integrate, we get

$$\frac{1}{2}w^2 = \frac{1}{\varepsilon^2}\Big(\frac{1}{4}u^4 - \frac{1}{2}u^2 + C\Big), \tag{2.8}$$

for some constant C. A homoclinic path, which is any phase path that joins an equilibrium points to itself, can only be associated with the saddle point ([23]) at $(\pm 1, 0)$. Since the phase paths approach to $(\pm 1, 0)$ only if $C = \frac{1}{4}$, we set $C = \frac{1}{4}$ in the equation (2.8) and solve for w to get:

$$\begin{aligned}
&\frac{1}{2}w^2 = \frac{1}{\varepsilon^2}\Big(\frac{1}{4}u^4 - \frac{1}{2}u^2 + \frac{1}{4}\Big)\\
\Rightarrow\ &\frac{1}{2}w^2 = \frac{1}{\varepsilon^2}\Big(\frac{1}{2}u^2 - \frac{1}{2}\Big)^2\\
\Rightarrow\ &\frac{1}{2}w^2 = \frac{1}{4\varepsilon^2}(u^2 - 1)^2\\
\Rightarrow\ &w^2 = \frac{1}{2\varepsilon^2}(u^2 - 1)^2\\
\Rightarrow\ &w = \pm\frac{\sqrt{2}}{2\varepsilon}(u^2 - 1).
\end{aligned} \tag{2.9}$$

Recall that $\dot{u} = w$, with $\dot{u} = \frac{du}{dx}$. Hence, the solution for the homoclinic paths can be found by integrating

$$w = \frac{du}{dx} = \pm\frac{\sqrt{2}}{2\varepsilon}(u^2 - 1).$$

By separating the variables (for simplicity, we omit $\pm$ here), we get

$$\int \frac{1}{\frac{\sqrt{2}}{2\varepsilon}(u^2 - 1)} du = \int dx.$$

Using the substitution $u = \tanh\left(\frac{\sqrt{2}}{2\varepsilon}y\right)$, and $du = \operatorname{sech}^2\left(\frac{\sqrt{2}}{2\varepsilon}y\right)\frac{\sqrt{2}}{2\varepsilon}dy$, we get

$$\begin{aligned} & \int \frac{\operatorname{sech}^2\left(\frac{\sqrt{2}}{2\varepsilon}y\right)\frac{\sqrt{2}}{2\varepsilon}}{\frac{\sqrt{2}}{2\varepsilon}(\tanh^2\left(\frac{\sqrt{2}}{2\varepsilon}y\right) - 1)} dy = \int dx \\ \Rightarrow \quad & \int \frac{\operatorname{sech}^2\left(\frac{\sqrt{2}}{2\varepsilon}y\right)\frac{\sqrt{2}}{2\varepsilon}}{\frac{\sqrt{2}}{2\varepsilon}(-\operatorname{sech}^2\left(\frac{\sqrt{2}}{2\varepsilon}y\right))} dy = \int dx \\ \Rightarrow \quad & -\int dy = \int dx \\ \Rightarrow \quad & y = -x + C_1 \end{aligned}$$

Hence, a solution is $u(x) = \tanh\left(\frac{\sqrt{2}}{2\varepsilon}x\right) = \frac{e^{\frac{\sqrt{2}}{\varepsilon}x}-1}{e^{\frac{\sqrt{2}}{\varepsilon}x}+1}$. The solution with $C_1 = 0$ changes rapidly from -1 to $+1$ near its zero at $x = 0$. Other choices for C_1 will produce translations of the solution in the x-direction.

As we can see in (2.7), as $x \to +\infty, u(x) \to +1$, and as $x \to -\infty, u(x) \to -1$. The width of the phase transition between ± 1 is determined by the value of ε. As ε gets bigger, the phase transition gets slower, whereas as ε gets smaller, the shape of the function approaches the Heaviside function.

If u is a two-variable real-valued function, $u : \mathbb{R}^2 \to \mathbb{R}$, using Maple, we can find one of the solutions for (2.6) as:

$$u(x, y) = \frac{e^{\frac{(x+y)}{\varepsilon}} - 1}{e^{\frac{(x+y)}{\varepsilon}} + 1}. \tag{2.10}$$

Similarly to the one-dimensional case, we can see in (2.10), as $x, y \to +\infty$, $u \to +1$, and as $x, y \to -\infty$, $u \to -1$. Also, the width of the phase transition between ± 1 is determined by the value of ε. As ε gets bigger, the phase transition gets slower, whereas as ε gets smaller,

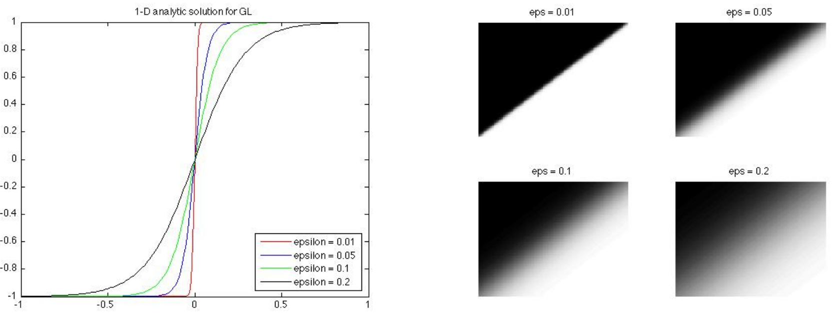

Figure 2.1: u is a Real-valued function Left: one-dimensional Right: two-dimensional

the shape of the function approaches the Heaviside function. As we mentioned above, the stationary points of the real-valued Ginzburg-Landau equation (2.6) are reached at $u = \pm 1$.

2.2.2 Inpainted images using the real-valued Ginzburg-Landau Equation

In this section, we will see inpainted images when we use the real-valued Ginzburg-Landau equation. We use the images in Figure 2.2 for these experiments.

The original Lincoln and girls images and their mask images are from [3] (They were originally from [4], but this is no longer available.), and the original parrot image and its mask image are from [5]. The images in the top row in Figure 2.2 are the original Lincoln's image and corresponding mask image. The original image was the sepia colour image in [4], but here we treat it as a grey scale image. The images in the middle row are the original girls' image and corresponding mask image. The mask image shows the region to be inpainted to be black. The images in the bottom row are the original parrot's image and corresponding mask image. The original image was the colour image in [4], but here we treat it as a grey scale image. Also, the mask image shows the region to be inpainted in black.

As we will discuss the details of inpainting processes later in Chapter 3, we only introduce the outline of the process here.

1. The images on the left-hand side of Figure 2.2 are the degraded images to be inpainted.
2. The images on the right-hand side of Figure 2.2 are the corresponding masks that define the region to be fixed.

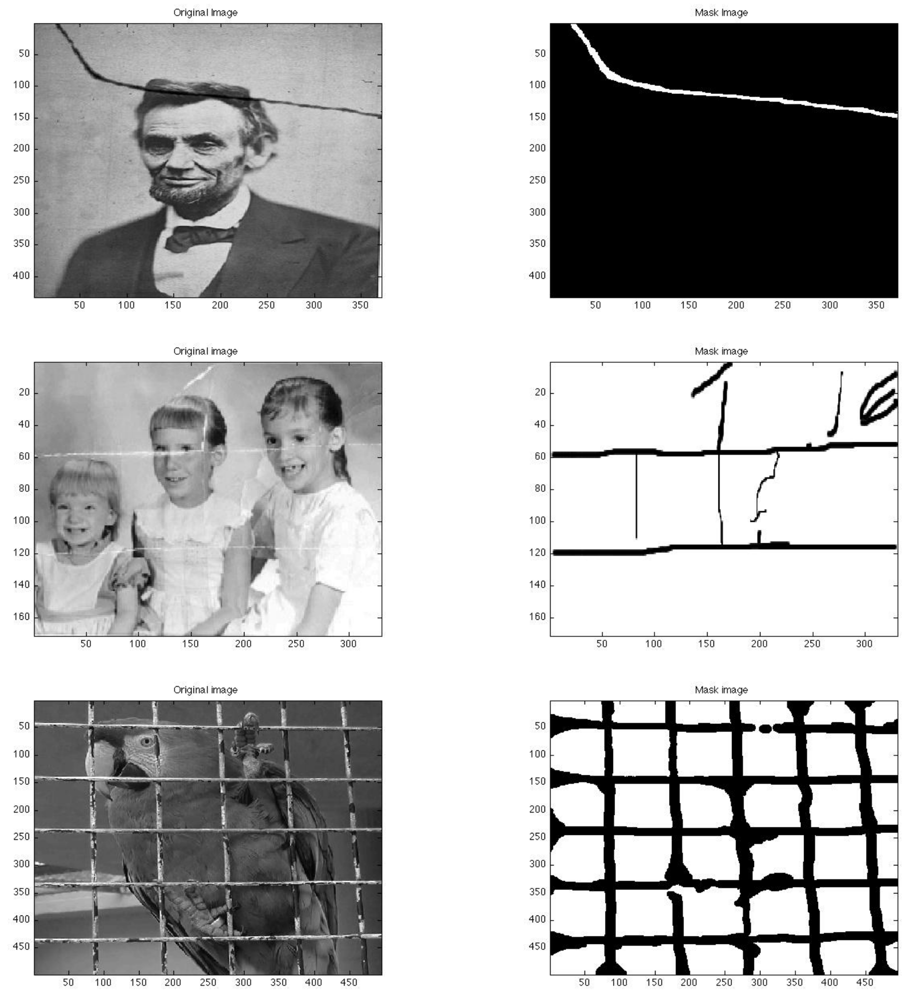

Figure 2.2: Top: Original Image (left) and Mask Image (right)

3. Scale the images between $[-1, 1]$.
4. Iterate the Ginzburg-Landau equation with the explicit discretization until the maximum number of iterations is reached.
   (a) For each iteration, the entire image is used to calculate the values of the Ginzburg-Landau equation.
   (b) After each iteration, only pixels defined by the mask are updated.
5. For these experiments, we use $\varepsilon = 0.1$, $\Delta t = \frac{\varepsilon^2}{4}$ and 1000 iterations.

Case 1 : First, we show what happens when we use the real-valued Ginzburg-Landau equation for inpainting.

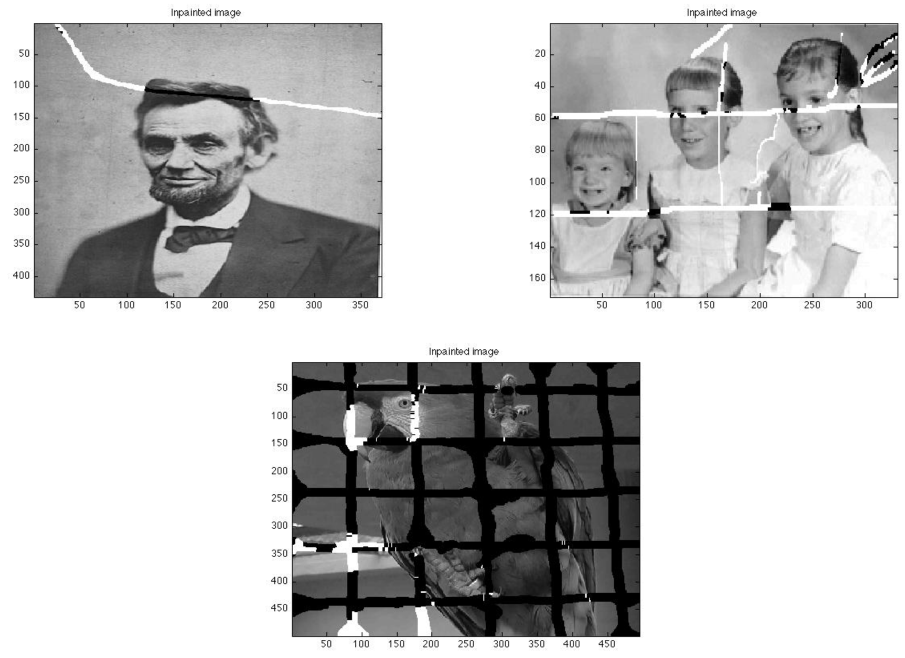

Figure 2.3: Inpainted with the Real-valued GLE

The results are shown in Figure 2.3. As expected, the regions to be inpainted are inpainted by only black or white. The regions where the outside region is darker are filled with black, and the regions where the outside region is brighter are filled with white. This is because the stationary points of the real-valued Ginzburg-Landau equation (2.6) are reached at $u = \pm 1$.

Case 2 : The inpainted images using only the diffusion term, and using the complex-valued Ginzburg-Landau Equation

We use the same process in **Case** 1, but here, we scale the image between $[-1, 1]$ and calculate the imaginary part as $\mathcal{I}(u^0) = \sqrt{1 - \mathcal{R}(u^0)^2}$ where $\mathcal{I}$ and $\mathcal{R}$ are the imaginary and real part of the image, respectively, and u^0 is the input image.
Now, we closely look at equation (2.6). Note that the first term of (2.6), Δu is to minimize the kinetic term in (1.2), $|-i\nabla u|$. This term tries to minimize the gradient of u, hence the image tends to blur. If we only use the first term of (2.6) Δu, for inpainting, we get the left image in Figure 2.4. We use the equation below with 1000 iterations to update the region to be inpainted,

$$\frac{\partial u}{\partial t} = \Delta u. \tag{2.11}$$

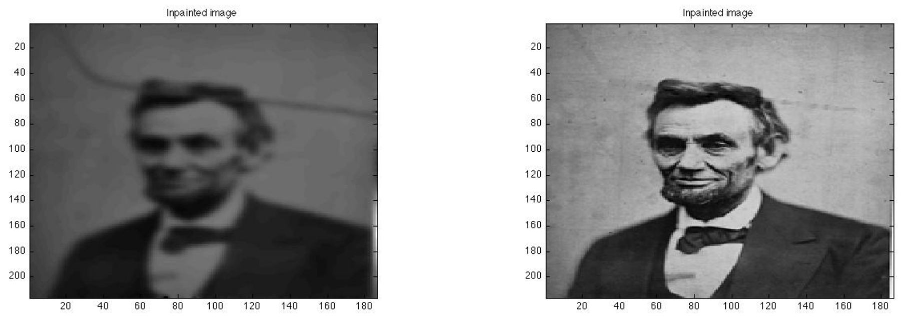

Figure 2.4: Left: Diffusion term only, Right: Via the complex-valued GLE

The left image in Figure 2.4 is the result when we use (2.11) over the entire image. As we expected, the image gets blurred since the term Δu tries to minimize the gradient of u.

The right image in Figure 2.4 is obtained when we use the complex-valued Ginzburg-Landau equation. We notice that the complex-valued Ginzburg-Landau equation does not lose the edges in the original image, hence the image does not get blurred. Also, since there is no phase transition developed with the complex-valued Ginzburg-Landau equation, the inpainted image does not look artificial. The phase transition is discussed in the next section.

2.2.3 $u(x)$ is a complex-valued function

If u is a complex-valued function, $u : \mathbb{R}^2 \to \mathbb{C}$, we seek the radial degree-one solution u ([30]) of

$$\Delta u + (1 - |u|^2)u = 0 \qquad \text{in} \quad \mathbb{R}^2 \tag{2.12}$$

setting $\varepsilon = 1$ in (2.6). The solution u is called a degree-one radial solution of (2.12), if u can be written in the form

$$u(r, \theta) = f(r)e^{i\theta}, \tag{2.13}$$

where (r, θ) are the polar coordinates in $\mathbb{R}^2$, and f is a scalar function $f : \mathbb{R}^+ \to \mathbb{R}^+$.
If we change the Laplacian to polar coordinates, we get ([6])

$$\Delta = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \theta^2}. \tag{2.14}$$

So, by using (2.14) we substitute (2.13) into (2.12), we get

$$\begin{aligned}
\Delta u + (1 - |u|^2)u &= f''e^{i\theta} + \frac{1}{r}f'e^{i\theta} + \frac{1}{r^2}i^2 e^{i\theta} f + (1 - |fe^{i\theta}|^2)fe^{i\theta} \\
&= (f'' + \frac{1}{r}f' - \frac{1}{r^2}f)e^{i\theta} + (1 - f^2)fe^{i\theta} \\
&= \Big[f'' + \frac{1}{r}f' - \frac{1}{r^2}f + (1 - f^2)f\Big]e^{i\theta} \\
&= 0.
\end{aligned}$$

Since $e^{i\theta} \neq 0$, f has to satisfy the second order ordinary differential equation,

$$f'' + \frac{1}{r}f' - \frac{1}{r^2}f + (1 - f^2)f = 0. \tag{2.15}$$

Then, according to [30] there exists a unique nonconstant degree-one radial solution u_0 of (2.15), and $f(r) \to 1$ as $r \to +\infty$.
This is explained also in [29], and the proof can be found in [21].

Therefore, if u is a complex-valued function, the minimum is attained when the values of u are on the complex unit circle, hence u does not develop phase transition.

2.2.4 Co-dimension one phase transition problem

Here, the co-dimension one phase transition problems of Ginzburg-Landau equation are discussed from [22].

The co-dimension one phase transition is the phase transition between co-dimension one objects. A co-dimension one object is represented by the zero isocontour of a function ϕ, and zero isocontours are points in a one dimensional domain, curves in a two dimensional domain, and surfaces in a three dimensional domain ([28]). We will look at a real-valued order parameter function u in $\mathbb{R}^1$ and $\mathbb{R}^2$.

1. Steady state problems

   According to [7], the order parameter is normally a quantity which is zero in one phase and non-zero in the other. In our case, the order parameter takes -1 in one phase and 1 in the other. We consider the minimization of the energy below over the domain Ω with the real-valued order parameter function u,

$$E_\varepsilon(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 + \frac{1}{2\varepsilon^2}(u^2-1)^2 dx. \tag{2.16}$$

   Here, ε is a small constant. The energy term, the double well potential $(u^2-1)^2$ will take the order parameter function u to be ± 1, and the the term $|\nabla u|^2$ is the approximation of the surface energy ([22]) which makes the transition of u smooth.

   - u is in $\mathbb{R}^1$

     In the one dimensional case, let $\Omega = (0,1)$, and let the boundary conditions be $u(0) = -1$, and $u(1) = 1$. Then the energy minimizer has to satisfy the following conditions with some fixed ε.

$$\begin{cases} -u_{xx} + \frac{1}{\varepsilon^2}(u^2-1)u = 0 \\ u(0) = -1, \quad u(1) = 1 \end{cases} \tag{2.17}$$

     Now, we multiply the differential equation (2.17) by u_x and integrate it from 0 to $x \in (0,1)$.

     If we multiply by u_x, we get:

$$\underbrace{-u_{xx}u_x}_{①} + \frac{1}{\varepsilon^2}\underbrace{u^3u_x}_{②} - \frac{1}{\varepsilon^2}\underbrace{uu_x}_{③} = 0. \tag{2.18}$$

By using the integration by parts, we get the following.

①:

$$\begin{aligned}\int_0^x u_{xx}u_x &= u_x^2\Big|_0^x - \int_0^x u_{xx}u_x \\ \Rightarrow 2\int_0^x u_{xx}u_x &= u_x^2(x) - u_x^2(0) \\ \Rightarrow \int_0^x u_{xx}u_x &= \frac{1}{2}(u_x^2(x) - u_x^2(0)) \end{aligned} \tag{2.19}$$

②:

$$\begin{aligned}\int_0^x u^3u_x &= u^4\Big|_0^x - 3\int_0^x u^3 \\ &= [u^4(x) - u^4(0)] - 3\frac{1}{4}[u^4(x) - u^4(0)] \\ &= \frac{1}{4}[u^4(x) - 1] \end{aligned} \tag{2.20}$$

③:

$$\begin{aligned}\int_0^x uu_x &= u^2\Big|_0^x - \int_0^x uu_x \\ \Rightarrow 2\int_0^x uu_x &= u^2(x) - u^2(0) \\ \Rightarrow \int_0^x uu_x &= \frac{1}{2}(u^2(x) - 1) \end{aligned} \tag{2.21}$$

Therefore, by using equations (2.19), (2.20) and (2.21), equation (2.18) becomes:

$$\begin{aligned}&-\frac{1}{2}(u_x^2(x) - u_x^2(0)) + \frac{1}{\varepsilon^2}\Big[\frac{1}{4}(u^4(x) - 1) - \frac{1}{2}(u^2(x) - 1)\Big] = 0 \\ &\Rightarrow -\frac{1}{2}(u_x^2(x) - u_x^2(0)) + \frac{1}{\varepsilon^2}\Big[\frac{1}{4}u^4(x) - \frac{1}{2}u^2(x) + \frac{1}{4})\Big] = 0 \\ &\Rightarrow -\frac{1}{2}(u_x^2(x) - u_x^2(0)) + \frac{1}{\varepsilon^2}\Big(\frac{1}{2}u^2(x) - \frac{1}{2}\Big)^2 = 0 \\ &\Rightarrow -u_x^2(x) + u_x^2(0) + \frac{1}{2\varepsilon^2}(u^2(x) - 1)^2 = 0 \\ &\Rightarrow u_x^2(0) + \frac{1}{2\varepsilon^2}(u^2(x) - 1)^2 = u_x^2(x). \end{aligned} \tag{2.22}$$

We look at the relation (2.22).

There are two cases:

(a) $u_x^2(0) \neq 0$:

Then we have $u_x^2(x) \neq 0$ in all $x \in (0,1)$.

This means that there is no horizontal tangent in the interval $(0,1)$. Hence, because of the boundary conditions at $u(0) = -1$ and $u(1) = 1$, the minimizer function u must be monotonically increasing.

(b) $u_x^2(0) = 0$ and

i. $|u|(x) \neq 1$ for all $x \in (0,1)$:

In this case, we have $u_x^2(x) \neq 0$ for all $x \in (0,1)$. Again, it means there is no horizontal tangent in the interval $(0,1)$. Hence, because of the boundary conditions, the minimizer function u must be monotonically increasing.

ii. $|u|(\xi) = 1$ for some $\xi \in (0,1)$:

In this case, we have $u_x^2(\xi) = 0$ for some $\xi \in (0,1)$.

Without loss of generality, we assume $u(\xi) = 1$. Then, we define a new function $\tilde{u}(x)$ such that

$$\tilde{u}(x) = \begin{cases} u & \text{if } x \in (0,\xi) \\ 1 & \text{if } x \in (\xi,1). \end{cases}$$

Then, $\tilde{u}$ has less energy than the minimizer function u, hence contradicting the fact that u is the minimizer function. As we can see the longer the interval of $u = 1$ is in $\tilde{u}(x)$, the smaller the energy of $\tilde{u}$. So, if we define a new function such that $u \equiv 1$ in all $u \in (0,1)$, then it has the smallest energy. But in this case, the function u can not satisfy the boundary conditions $u(0) = -1$ and $u(1) = 1$ at the same time. Hence, we conclude that $u(x) < 1$ for all $x \in (0,1)$.

The symmetric argument holds for $u(\xi) = -1$. Hence, $|u|(x) \neq 1$ in all $x \in (0,1)$.

Therefore, the energy minimizer u_ε does not oscillate between -1 and 1. Hence, we have shown that there exists a unique $\xi_0 \in (0,1)$ such that as $\varepsilon \to 0$,

$$\begin{aligned} u_\varepsilon(x) &\to -1 \quad \text{when} \quad x \in (0,\xi_0) \\ u_\varepsilon(x) &\to \ \ 1 \quad \text{when} \quad x \in (\xi_0,1). \end{aligned}$$

- u is in $\mathbb{R}^2$

In the two dimensional case, if we let the domain be $\Omega = (0,1) \times (0,1)$, then we need continuity at all edges of a rectangle formed by $(0,0)$, $(1,0)$, $(0,1)$, and $(1,1)$. Hence, a Dirichlet boundary condition does not work anymore. So, we impose a Neumann boundary condition. Then, in [22], the energy minimizer has to satisfy the following conditions with some fixed ε,

$$\begin{cases} -\Delta u + \frac{1}{\varepsilon^2}(u^2-1)u = c \\ \int_\Omega u(x) = m \int_\Omega dx \\ \frac{\partial u}{\partial \mathbf{n}} = 0 \end{cases} \tag{2.23}$$

with some constant $m \in (-1,1)$ and some unknown constant c.
In this case, the analogous result to the one dimensional case is : as $\varepsilon \to 0$, the minimizer of (2.16) tends to a function u_0 almost everywhere such that Ω is divided into two domains Ω_1 and Ω_2 and

$$u_0(x) = \begin{cases} -1 & x \in \Omega_1 \\ \ \ 1 & x \in \Omega_2 \end{cases} \qquad ([22]).$$

Here, we do not prove this, instead, we will show these conditions on $u_0(x)$ experimentally by plotting the function u which is a solution of the equation

$$\Delta u + \frac{1}{\varepsilon^2}(1-|u|^2)u = 0. \tag{2.24}$$

We have confirmed with Maple that the two-variable real-valued function, $u : \mathbb{R}^2 \to \mathbb{R}$,

$$u(x,y) = \frac{e^{\frac{(x+y-1)}{\varepsilon}} - 1}{e^{\frac{(x+y-1)}{\varepsilon}} + 1} \tag{2.25}$$

satisfies the equation (2.24). We plot the function (2.25) in the domain of $[0,1] \times [0,1]$ for $\varepsilon = 0.005, 0.01, 0.05, 0.1$, and see the following results in Figure 2.5.

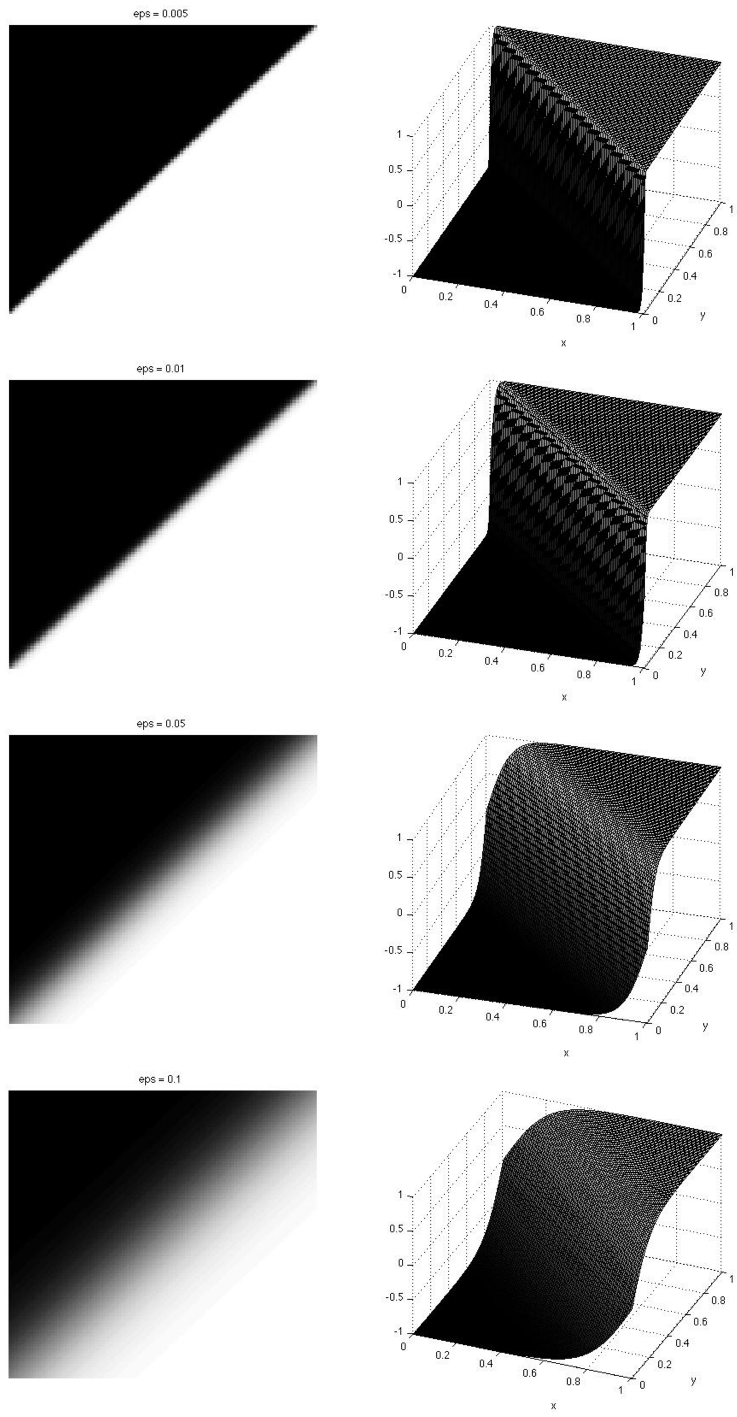

Figure 2.5: Plots of equation (2.25) with $\varepsilon = 0.005$, 0.01, 0.05 and 0.1

The pictures on the left hand side are the images of domains Ω_1 and Ω_2, and the pictures on the right hand side are the function in (2.25) . As we can see, clearly $u_0(x, y) = 1$ for $(x, y) \in \Omega_1$ and $u_0(x, y) = -1$ for $(x, y) \in \Omega_2$. The regions Ω_1 and Ω_2 are separated by the line $y = 1 - x$. We observe that as the value of ε gets closer to zero, the boundary of Ω_1 and Ω_2 gets sharper in the left hand side of Figure 2.5. Also, as the value of ε gets smaller, the slope of the boundary between Ω_1 and Ω_2 gets steeper in the right hand of Figure 2.5.

2. Long time behavior

In [22], the long-time behaviour of the minimizer of (2.16) is investigated. The main result is that as $t \to \infty$, the solution of the following problem

$$\begin{cases} u_t - \Delta u + (u^2-1)u = c(t), & \text{in } \Omega \times (0,\infty) \\ \int_\Omega u(x,t)dx = a = m\int_\Omega dx, & \text{on } (0,\infty) \\ \frac{\partial u}{\partial \mathbf{n}} = 0, & \text{on } \partial\Omega \times (0,\infty) \\ u(x,0) = u_0(x), & \text{in } \Omega, \end{cases} \tag{2.26}$$

where $c(t)$ is a function of time, t only, a is a constant independent of time, t, $u_0(x) \in H^1(\Omega)$ and $|u_0(x)| \leq 1$ for almost every $x \in \Omega$,
converges to the set of solutions of the following problem,

$$\begin{cases} -\Delta u + (u^2-1)u = c, \\ \int_\Omega u(x,t)dx = a = m\int_\Omega dx, \\ \frac{\partial u}{\partial \mathbf{n}} = 0, \end{cases} \tag{2.27}$$

for some suitable constant c.

We integrate the equation (2.26) by using the associated conditions. Then we get,

$$\underbrace{\int_\Omega u_t dx}_{①} - \underbrace{\int_\Omega \Delta u dx}_{②} + \underbrace{\int_\Omega (u^2-1)u dx}_{③} = \underbrace{\int_\Omega c(t) dx}_{④}. \tag{2.28}$$

The left side:

①: $\int_\Omega u_t dx = a_t = 0$

Since $\int_\Omega u(x,t)dx = a$, where a is a constant independent of time,

$\int_\Omega u_t dx = \frac{d}{dt}\Big(\int_\Omega u(x,t)dx \Big) = \frac{d}{dt}(a) = 0.$

②: $-\int_\Omega \Delta u dx = 0$
Since by the boundary condition, we have $\frac{\partial u}{\partial \mathbf{n}} = \nabla u \cdot \mathbf{n}(x) = 0$.
And by the divergence theorem, $\int_\Omega \mathrm{div}\nabla u \, d\Omega = \int_{\partial\Omega} \nabla u \cdot \mathbf{n} \, dS = 0$,
where $\mathrm{div}\nabla u = \Delta u$.

Hence, we get,

$$\begin{aligned} &\int_\Omega (u^2-1)u dx = \int_\Omega c(t) dx \\ &\Rightarrow \int_\Omega (u^2-1)u dx = c(t)\int_\Omega dx \\ &\Rightarrow c(t) = \frac{\int_\Omega (u^2-1)u dx}{\int_\Omega dx}. \end{aligned} \tag{2.29}$$

Also, according to [22], $E(u)(t) = \frac{1}{2}\int_\Omega |\nabla u|^2 + \frac{1}{2}(u^2-1)^2 dx$ is the Lyapunov functional for the problem (2.26).
Hence, we can conclude ([22]) that for any given smooth initial data u_0, there exists a constant $C > 0$ independent of time t, such that

$$\begin{aligned} &||u||_{H^1(\Omega)}(t) \leq C, \\ &|c(t)| \leq C. \end{aligned}$$

Using standard parabolic PDE theory ([22]), we have

$$||u||_{H^2(\Omega)}(t) \leq C.$$

Using standard ω-convergence theory ([22]), we have that as $t \to \infty$, $u(x,t)$ converges to the set of solutions of (2.27). (The details of the proof for this are found in [22] .)

2.2.5 Co-dimension two phase transition problem

In this section, the co-dimension two phase transition problems of the Ginzburg-Landau equation are discussed from [22].

Here, we represent co-dimension two geometry as the intersection of the zero isocontour of a function ϕ_1 with the zero isocontour of another function ϕ_2. The intersection of points that are zero isocontours of co-dimension one objects in $\mathbb{R}^1$ is the empty set in co-dimension two objects. The intersection of curves that are zero isocontours of co-dimension one objects in $\mathbb{R}^2$ is the point in co-dimension two objects. And, the intersection of surfaces that are zero isocontours of co-dimension one objects in $\mathbb{R}^3$ is the curve in co-dimension two objects ([28]). Co-dimension two phase transition domains are also called vortices, and as explained, they are points in $\mathbb{R}^2$ and curves in $\mathbb{R}^3$. We will look at a complex-valued order parameter function u in $\mathbb{R}^2$.

1. Steady state problems

   To describe co-dimension two phase transition, we need two co-dimension one objects u_1 and u_2, which are real-valued functions. Then, we can minimize the energy of

   $$\frac{1}{2}\int_\Omega |\nabla u_1|^2 + |\nabla u_2|^2 + \frac{1}{2\varepsilon^2}((u_1^2-1)^2 + (u_2^2-1)^2)dx.$$

   Since each of u_1 and u_2 has a co-dimension one phase transition region, the intersection of these two co-dimension one regions might form a co-dimension two phase transition region. But, since both function u_1 and u_2 take the value of ± 1 near the co-dimension one transition regions, we would have four phases $(1,1),(1,-1),(-1,1)$ and $(-1,-1)$ near the intersection of co-dimension two phase transition region. As we can see this is rather complicated, hence, we introduce a complex-valued order parameter function u ([22]).

   Then, the energy is:

   $$\frac{1}{2}\int_\Omega |\nabla u|^2 + \frac{1}{2\varepsilon^2}(|u|^2-1)^2 dx, \tag{2.30}$$

   and the corresponding Euler equation is

   $$-\Delta u^2 + \frac{1}{\varepsilon^2}(|u|^2-1)u = 0, \tag{2.31}$$

   for a complex-valued function u.

   From the Euler equation (2.31), the minimizer of the energy term $(|u|^2-1)^2$ takes $|u|=1$. But, since it is on the unit circle, there is no transition phase developed like for the real-valued function where the values of ± 1 are isolated points.

The other extreme value of the energy term $(|u|^2-1)^2$ occurs at u equals 0. As we can write $u(r,\theta)=g(r)e^{i\theta}$ for a scalar function $g \geq 0$, we get

$$|u(r,\theta)| = |g(r)e^{i\theta}| = |g(r)||e^{i\theta}| = g(r).$$

Hence $f(u) = (|u|^2-1)^2 = (g^2-1)^2$. If we take a first derivative, we get,

$$\begin{aligned} f^{'}(u) &= 2(g^2-1)2g \\ &= 4g^3-4g, \end{aligned}$$

and therefore, $f^{'}(0)=0$. If we take a second derivative, we get,

$$f^{''}(u) = 12g^2-4.$$

Thus, at $u=0$, i.e. $g=0$, $f^{''}(0)=-4<0$. Therefore, 0 is a local maximum.
As the equation is unstable at $u=0$, the order parameter function u does not take the value of zero easily. Once u takes the value of 0, it leaves the value as soon as possible. As a consequence, the order parameter u will minimize the region where it has to take the value zero because of instability ([22]).

Also, the article [12] says that under certain conditions on u_0, a minimizer u_ε of $E_\varepsilon(u)$, where

$$E_\varepsilon(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 + \frac{1}{4\varepsilon^2}\int_\Omega (|u|^2-1)^2, \tag{2.32}$$

converges to u_0 as $\varepsilon \to 0$.

2. Long time behaviour
According to [22], a similar argument as discussed in Section **2.2.4** can be applied to the complex-valued order parameter u.
Let $u(x,t)$ be the solution of

$$u_t = \Delta u - \frac{1}{\varepsilon^2}(|u|^2-1)u.$$

Then, the distance measured by H^1 norms between u and the set of solutions of

$$-\Delta u + \frac{1}{\varepsilon^2}(|u|^2-1)u = 0$$

approaches zero as $t \to \infty$ under appropriate boundary conditions and restrictions. ([22]).

Chapter 3

Implementation of the Algorithm

In this chapter, we present an algorithm for inpainting based on the Ginzburg-Landau equation for a grey scale image.

3.1 Implementation

1. What we need for the implementation of the method

   - an image u

2. What the user must define for the inpainting

   - an inpainting domain Ω (i.e. an area to be inpainted)

3. How to implement the algorithm

   (1) Define u^0 by scaling u, so that the real part of u^0 is defined as $\mathcal{R}(u^0) \in [-1, 1]$.
   The imaginary part of u^0 is selected by $\mathcal{I}(u^0) = \sqrt{1 - \mathcal{R}(u^0)^2}$
   so that $u^0 : D \to \mathbb{C}$, and $|u^0(x)| = 1$ for all $x \in D$.

   (2) Set the area to be inpainted, Ω, to be 0 as initial value.
   i.e. $u^0|_\Omega = 0$.

   (3) Use the steepest descent method to find the solution of (2.6) numerically

   $$u_t = \Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u \tag{3.1}$$

   until a stopping criteria is satisfied.
   We update only the values inside of the area to be inpainted, Ω.

3.2 Explicit Discretization

The state of minimal energy of the Ginzburg-Landau equation satisfies the Euler equation of $F(u, \nabla u)$:

$$\Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u = 0. \tag{2.6}$$

To find the solution of (2.6) with Dirichlet boundary condition numerically, we use the steepest descent method to solve the differential equation

$$u_t = \Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u \tag{3.1}$$

up to the stationary point in time. We discretize the equation (3.1) in time and space according to [24].

3.2.1 Time discretization

We discretize time by using the forward Euler method. We replace $u_t = f$ by

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = f(u_{i,j}^n). \tag{3.2}$$

And when we solve (3.2) explicitly for $u_{i,j}^{n+1}$ in terms of $u_{i,j}^n$, we get

$$u_{i,j}^{n+1} = u_{i,j}^n + \Delta t f(u_{i,j}^n). \tag{3.3}$$

If we let $u_{i,j}^0$ be the initial data, from $u_{i,j}^0$, we compute $u_{i,j}^1$, then $u_{i,j}^2$ up to the stationary point. This is called a *time marching method.* ([24])

3.2.2 Space discretization

Now we discretize the equation (3.1) in space. When we discretize the right hand side of the equation (3.1), we use the centred finite difference.

$$\Delta u_{i,j} = \frac{1}{\Delta x^2}\underbrace{\Big[u_{i-1,j}^n - 2u_{i,j}^n + u_{i+1,j}^n\Big]}_{x-direction} + \frac{1}{\Delta y^2}\underbrace{\Big[u_{i,j-1}^n - 2u_{i,j}^n + u_{i,j+1}^n\Big]}_{y-direction},$$

if we let $\Delta x = \Delta y = h$,

$$\Delta u_{i,j} = \frac{1}{h^2}\Big[u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n - 4u_{i,j}^n\Big]. \tag{3.4}$$

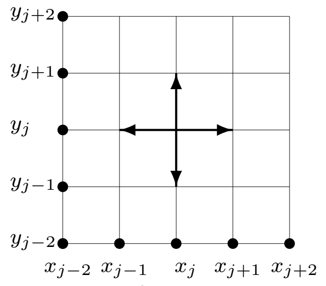

Figure 3.1: The 5-point stencil for the Laplacian about the point (i, j)

To simplify the notation, we assume that $h = 1$. Then, the above equation (3.1) can be written as:

$$\begin{aligned} u_{i,j}^{n+1} &= u_{i,j}^n + \Delta t[u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n - 4u_{i,j}^n] \\ &+ \frac{\Delta t}{\varepsilon^2}\Big[(1 - |u_{i,j}^n|^2)u_{i,j}^n\Big]. \end{aligned}$$

3.2.3 Convergence of Explicit method

In this section, we will check the convergence of the explicit finite difference method. Since the convergence of a numerical method for a linear PDE requires consistency and stability, we prove the consistency and the stability of the method.

1. Consistency of the explicit method

   Let the discrete system

$$\begin{cases} F_{\Delta x,\Delta y,\Delta t}(u_{\Delta x,\Delta y}^{\Delta t}) = f_{\Delta x,\Delta y}, \qquad x, y \in \Omega_{\Delta x,\Delta y}, \qquad t \in T_{\Delta t} \\ \text{Boundary conditions} \\ \text{Initial conditions} \end{cases}$$

   be used as a finite difference discretization for the parabolic problem

$$\begin{cases} u_t = \mathcal{L}(u) + f, \qquad x, y \in \Omega, \qquad t \in (0, T] \\ \text{Boundary conditions} \\ \text{Initial conditions.} \end{cases}$$

   Here, $\Omega_{\Delta x,\Delta y}$ is the spatial grid with mesh size Δx in x-direction and Δy in y-direction, and $T_{\Delta t}$ is the time steps with $T_{\Delta t} = \{0, \Delta t, 2\Delta t, \cdots, T\}$.

The scheme is said to be consistent with the differential equation and boundary conditions if

$$\tau_{i,j}^{n} := [F_{\Delta x,\Delta y,\Delta t}(u(x_i, y_j, t_n)) - f_{\Delta x,\Delta y}] - [u_t - \mathcal{L}(u) - f]\Big|_{x_i,y_j,t_n}$$

approaches zero as Δx, Δy and $\Delta t \to 0$.
We calculate the truncation error $\tau_{i,j}^{n}$ by using the Taylor series expansion for $u(x_i, y_j, t_n)$. Let $u_{i,j}^{n}$ be the discretized version of $u(x_i, y_j, t_n)$. Our equation

$$u_t = \Delta u + f(u) \tag{3.5}$$

can be written in the finite difference scheme as:

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = \frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{\Delta x^2} + \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2} + f(u_{i,j}^{n}). \tag{3.6}$$

Hence, the truncation error can be calculated as:

$$\begin{aligned}\tau_{i,j}^{n+1} := &\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} - \frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{\Delta x^2} - \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2} - f(u_{i,j}^{n})\\ &- (u_t - \Delta u - f(u))\Big|_{x_i,y_j,t_n},\end{aligned}$$

using Taylor series expansion,

$$
\begin{aligned}
\tau_{i,j}^{n+1} := \frac{1}{\Delta t}\Big[&u(x_i,y_j,t_n) + \Delta t\, u_t(x_i,y_j,t_n) + \frac{\Delta t^2}{2}u_{tt}(x_i,y_j,t_n) + \cdots - u(x_i,y_j,t_n)\Big] \\
- \frac{1}{\Delta x^2}\Big[&u(x_i,y_j,t_n) + \Delta x\, u_x(x_i,y_j,t_n) + \frac{\Delta x^2}{2}u_{xx}(x_i,y_j,t_n) \\
&+ \frac{\Delta x^3}{6}u_{xxx}(x_i,y_j,t_n) + \frac{\Delta x^4}{24}u_{xxxx}(x_i,y_j,t_n)\cdots \\
&- 2\,u(x_i,y_j,t_n) \\
&+ u(x_i,y_j,t_n) - \Delta x\, u_x(x_i,y_j,t_n) + \frac{\Delta x^2}{2}u_{xx}(x_i,y_j,t_n) \\
&- \frac{\Delta x^3}{6}u_{xxx}(x_i,y_j,t_n) + \frac{\Delta x^4}{24}u_{xxxx}(x_i,y_j,t_n)\cdots\Big] \\
- \frac{1}{\Delta y^2}\Big[&u(x_i,y_j,t_n) + \Delta y\, u_y(x_i,y_j,t_n) + \frac{\Delta y^2}{2}u_{yy}(x_i,y_j,t_n) \\
&+ \frac{\Delta y^3}{6}u_{yyy}(x_i,y_j,t_n) + \frac{\Delta y^4}{24}u_{yyyy}(x_i,y_j,t_n)\cdots \\
&- 2\,u(x_i,y_j,t_n) \\
&+ u(x_i,y_j,t_n) - \Delta y\, u_y(x_i,y_j,t_n) + \frac{\Delta y^2}{2}u_{yy}(x_i,y_j,t_n) \\
&- \frac{\Delta y^3}{6}u_{yyy}(x_i,y_j,t_n) + \frac{\Delta y^4}{24}u_{yyyy}(x_i,y_j,t_n)\cdots\Big] \\
&- f(u(x_i,y_j,t_n)) \\
- \big(&u_t(x_i,y_j,t_n) - u_{xx}(x_i,y_j,t_n) - u_{yy}(x_i,y_j,t_n) - f(u(x_i,y_j,t_n))\big) \\
= \frac{\Delta t}{2}&u_{tt}(x_i,y_j,t_n) - \frac{\Delta x^2}{12}u_{xxxx}(x_i,y_j,t_n) - \frac{\Delta y^2}{12}u_{yyyy}(x_i,y_j,t_n). \qquad (3.7)
\end{aligned}
$$

Therefore,

$$|\tau_{i,j}^{n}| \le |\frac{\Delta t}{2}u_{tt}(x_i,y_j,t_n)| + |\frac{\Delta x^2}{12}u_{xxxx}(x_i,y_j,t_n)| + |\frac{\Delta y^2}{12}u_{yyyy}(x_i,y_j,t_n)|$$

or

$$\lim_{\Delta x,\Delta y,\Delta t\to 0} |\tau_{i,j}^{n}| = 0.$$

Hence, the scheme is consistent.

2. Stability of the explicit method

   In order to show the stability, we use the discrete von Neumann criterion for stability ([31]). To apply the discrete von Neumann criterion to a two dimensional problem, we consider a discrete Fourier mode for the problem of the form

$$u^n_{j,k} = \xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y}, \tag{3.8}$$

   where $i^2 = -1$ and p, q are the wave numbers for x and y respectively. Then, we get

$$u^{n+1}_{j,k} = \xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y}, \tag{3.9}$$

   hence,

$$\frac{u^{n+1}_{j,k}}{u^n_{j,k}} = \frac{\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y}}{\xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y}} = \xi. \tag{3.10}$$

   The von Neumann condition is that the amplification factor $|\xi| \le 1$. Since the von Neumann condition guarantees the stability of a finite difference method, we will show that $|\xi| \le 1$ for this scheme.

Equation (3.1) includes the nonlinear term $\frac{1}{\varepsilon^2}(1-|u|^2)u$. Since a complex-valued solution u of (3.1) will have an absolute value of 1 almost everywhere except the region to be inpainted, the nonlinear term is zero almost everywhere except the region to be inpainted. Therefore, if ε is large enough, we can ignore this term. As $1-|u|^2 \le 1$ is always true in (3.1), we see the relationship between the spacial step size h and the size of ε in the following way.

Since $1-|u|^2 \le 1$ in (3.1), we get

$$\frac{u^{n+1}_{i,j} - u^n_{i,j}}{\Delta t} \le \frac{u^n_{i+1,j} - 2u^n_{i,j} + u^n_{i-1,j}}{\Delta x^2} + \frac{u^n_{i,j+1} - 2u^n_{i,j} + u^n_{i,j-1}}{\Delta y^2} + \frac{1}{\varepsilon^2}u^n_{i,j}. \tag{3.11}$$

And we plug the expression in (3.8) into it. This becomes

$$\begin{aligned}
&\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} \le \xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y} \\
&+ \frac{\Delta t}{\Delta x^2}[\xi^n e^{i(j+1)p\pi\Delta x + ikq\pi\Delta y} - 2\xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y} + \xi^n e^{i(j-1)p\pi\Delta x + ikq\pi\Delta y}] \\
&+ \frac{\Delta t}{\Delta y^2}[\xi^n e^{ijp\pi\Delta x + i(k+1)q\pi\Delta y} - 2\xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y} + \xi^n e^{ijp\pi\Delta x + i(k-1)q\pi\Delta y}] \\
&+ \frac{\Delta t}{\varepsilon^2}\xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y}.
\end{aligned} \tag{3.12}$$

By using the trigonometric identities $\cos(z) = \frac{e^{iz}+e^{-iz}}{2}$ and $\cos(2x) = 1 - 2\sin^2(x)$, we simplify the equation (3.12), and we get

$$\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} \leq$$
$$\xi^{n} e^{ijp\pi\Delta x + ikq\pi\Delta y} \Big[1 + \frac{\Delta t}{\Delta x^2}(e^{ip\pi\Delta x} - 2 + e^{-ip\pi\Delta x}) + \frac{\Delta t}{\Delta y^2}(e^{iq\pi\Delta y} - 2 + e^{-iq\pi\Delta y}) + \frac{\Delta t}{\varepsilon^2} \Big].$$

Hence,

$$\begin{aligned}
\xi &\leq 1 + \frac{\Delta t}{\Delta x^2}(2\cos(p\pi\Delta x) - 2) + \frac{\Delta t}{\Delta y^2}(2\cos(q\pi\Delta y) - 2) + \frac{\Delta t}{\varepsilon^2} \\
&\leq 1 + 2\frac{\Delta t}{\Delta x^2}(\cos(p\pi\Delta x) - 1) + 2\frac{\Delta t}{\Delta y^2}(\cos(q\pi\Delta y) - 1) + \frac{\Delta t}{\varepsilon^2} \\
&\leq 1 + 2\frac{\Delta t}{\Delta x^2}\Big[1 - 2\sin^2(\frac{p\pi\Delta x}{2}) - 1\Big] + 2\frac{\Delta t}{\Delta y^2}\Big[1 - 2\sin^2(\frac{q\pi\Delta y}{2}) - 1\Big] + \frac{\Delta t}{\varepsilon^2} \\
&\leq 1 - 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{p\pi\Delta x}{2}) - 4\frac{\Delta t}{\Delta y^2}\sin^2(\frac{q\pi\Delta y}{2}) + \frac{\Delta t}{\varepsilon^2}. \qquad (3.13)
\end{aligned}$$

Since $|\sin(x)| \leq 1$ for any argument x, we have $|\xi| \leq 1$ as long as

$$\begin{aligned}
|1 - \Big(4\frac{\Delta t}{\Delta x^2} + 4\frac{\Delta t}{\Delta y^2}\Big) + \frac{\Delta t}{\varepsilon^2}| &\leq 1 \\
\Rightarrow -1 \leq 1 - \Big(4\frac{\Delta t}{\Delta x^2} + 4\frac{\Delta t}{\Delta y^2}\Big) + \frac{\Delta t}{\varepsilon^2} &\leq 1. \qquad (3.14)
\end{aligned}$$

Then, there are two cases to consider.

Case 1: $1-\left(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\right)+\frac{\Delta t}{\varepsilon^2}\leq 1$
Then, we get,

$$\begin{aligned}-\Big(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\Big)+\frac{\Delta t}{\varepsilon^2}&\leq 0\\ \frac{\Delta t}{\varepsilon^2}&\leq\Big(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\Big)\\ \text{If we let } \Delta x=\Delta y=h\text{, we get}&\\ \frac{\Delta t}{\varepsilon^2}&\leq 8\frac{\Delta t}{h^2}\\ \text{Since } \Delta t\neq 0\text{, we get,}&\\ \frac{1}{\varepsilon^2}&\leq\frac{8}{h^2}\\ \varepsilon^2&\geq\frac{h^2}{8}\\ \varepsilon&\geq h\sqrt{\frac{1}{8}}.\end{aligned}$$

Therefore, ε is not allowed to be much smaller than h, which is a reasonable restriction. In particular, if we let $h=1$, then $\varepsilon\geq\sqrt{\frac{1}{8}}\approx 0.3536$ (a pessimistic bound). As we mentioned above, we can ignore the nonlinear term if ε is large enough. Since if $\Delta x=\Delta y=1$, ε has to be approximately $\varepsilon\geq 0.36$, we have shown that we can ignore this term in the stability analysis.
If we ignore the nonlinear term, i.e. $\frac{\Delta t}{\varepsilon^2}$ in Case 1, since $1-\left(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\right)\leq 1$ is always true, we check the left side inequality in (3.14) as Case 2 (without the nonlinear term).

Case 2: $-1\leq 1-\left(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\right)$
Then, we get,

$$\begin{aligned}&-1\leq 1-\Big(4\frac{\Delta t}{\Delta x^2}+4\frac{\Delta t}{\Delta y^2}\Big)\\ &\Rightarrow\ 4\Big(\frac{\Delta t}{\Delta x^2}+\frac{\Delta t}{\Delta y^2}\Big)\leq 2\\ &\Rightarrow\ \frac{\Delta t}{\Delta x^2}+\frac{\Delta t}{\Delta y^2}\leq\frac{1}{2}.\end{aligned}\tag{3.15}$$

Therefore, for the scheme to be stable, we need condition (3.15) and ε has to satisfy the condition of $\varepsilon\geq h\sqrt{\frac{1}{8}}$ when we assume $\Delta x=\Delta y=h$.

3.3 Implicit Discretization

The implicit discretization is more complicated than the explicit one since the equation is not linear. As before, we want to discretize

$$u_t = \Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u. \tag{3.1}$$

We discretize the equation (3.1) in time and space according to [24].

3.3.1 Fully Implicit discretization

We discretize time by using the backward Euler method. We replace $u_t = f$ with:

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = f(u_{i,j}^{n+1})$$

or

$$u_{i,j}^{n+1} = u_{i,j}^{n} + \Delta t f(u_{i,j}^{n+1}). \tag{3.16}$$

Note that f is evaluated at the new time step. In the backward Euler method, we have to solve for $u_{i,j}^{n+1}$ in (3.16) and $f(u)$ is a nonlinear function. If we let the function $G(u^{n+1}) = u^{n+1} - \Delta t f(u^{n+1}) - u^n$ and look for a zero of $G(u^{n+1})$, then it can be approximated by Newton's method.

3.3.2 Newton's method

Here, we consider the one variable real-valued function. Hence, u is a vector with the size of m-by-1. We discretize the nonlinear problem

$$u_t = \Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u \tag{3.1}$$

using the following approach.

Let $F(u) = \frac{1}{\varepsilon^2}(1-|u|^2)u$, and rewrite the equation (3.1) by letting $\Delta t = k$ and $\Delta x = h$:

$$\frac{u_i^{n+1} - u_i^n}{k} = \frac{1}{h^2}(u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}) + F(u_i^{n+1})$$

or

$$u_i^{n+1} - u_i^n = \frac{k}{h^2}(u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}) + kF(u_i^{n+1})$$

or

$$u_i^{n+1} - \frac{k}{h^2}(u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}) - kF(u_i^{n+1}) - u_i^n = 0. \tag{3.17}$$

Now, this is a nonlinear system of equations of the form $G(u^{n+1}) = 0$. Hence, we use some iterative method such as Newton's method to solve for u^{n+1}. If u^n is the approximation to u in step n, then Newton's method is derived from Taylor series expansion about u^{n+1}, and we get:

$$G(u^{n+1}) = G(u^n) + G^{'}(u^n)(u^{n+1} - u^n) + \text{higher order terms}.$$

By setting $G(u^{n+1}) = 0$ and dropping the higher order terms, we get:

$$0 = G(u^n) + G^{'}(u^n)(u^{n+1} - u^n).$$

Solving for u^{n+1}, we get

$$u^{n+1} = u^n - [G^{'}(u^n)]^{-1}G(u^n),$$

(provided the inverse exists), which is Newton's method.
In our case,

$$G(u_i^{n+1}) = u_i^{n+1} - \frac{k}{h^2}(u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}) - kF(u_i^{n+1}) - u_i^n,$$

with $F(u_i^{n+1}) = \frac{1}{\varepsilon^2}(1-|u_i^{n+1}|^2)u_i^{n+1} = \frac{1}{\varepsilon^2}(u_i^{n+1} - u_i^{n+1}|u_i^{n+1}|^2)$, and since the discretization matrix (which is an m-by-m matrix) for the one dimensional Laplacian looks like

$$A = \begin{pmatrix} -2 & 1 & 0 & \dots & \\ 1 & -2 & 1 & 0 & \dots \\ 0 & 1 & -2 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix},$$

we can write $G(u_i^{n+1})$ as

$$G(u_i^{n+1}) = (I - \frac{k}{h^2}A)u_i^{n+1} - \frac{k}{\varepsilon^2}(u_i^{n+1} - u_i^{n+1}|u_i^{n+1}|^2) - u_i^n,$$

where I is an m-by-m identity matrix. Hence,

$$G^{'}(U^{n+1}) = (I - \frac{k}{h^2}A) - \frac{k}{\varepsilon^2}(I - J^{n+1})$$

where J is the m-by-m Jacobian matrix

$$J = \begin{pmatrix} 3|u_1^{n+1}|^2 & 0 & 0 & \dots & \\ 0 & 3|u_2^{n+1}|^2 & 0 & 0 & \dots \\ 0 & 0 & 3|u_3^{n+1}|^2 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \\ 0 & 0 & 0 & 0 & 3|u_m^{n+1}|^2 \end{pmatrix}.$$

So far, we looked at the case when u is an m-by-1 vector. Notice that since the image is usually represented as an m-by-m matrix instead of a vector, if we use the fully implicit method to solve the Ginzburg-Landau equation, we need m^2-by-m^2 matrices for a two-dimensional Laplacian matrix, an identity matrix and a Jacobian matrix. We will see the result of this method in Chapter 4, but if the size of an image is 350-by-350, then these matrices are of size 122500-by-122500. This is too large to fit into RAM, so we have to use matlab function `sparse()` to manipulate the matrices. Hence, it takes a long time to compute the solution.

3.3.3 Semi-Implicit method

To avoid the use of the large Jacobian matrix of J, we will try the semi-implicit method here. Again, we consider the equation:

$$u_t = \Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u. \tag{3.1}$$

As before in the fully implicit scheme, we discretize Δu with u^{n+1}, but for the nonlinear term $F(u)$, we discretize with u^n and u^{n+1}. More precisely, we let the discretization be

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2} + \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{\Delta y^2} + \frac{1}{\varepsilon^2}(1 - |u_{i,j}^n|^2)u_{i,j}^{n+1}. \tag{3.18}$$

If we let the two-dimensional discretization matrix for the Laplacian be D, we get,

$$\frac{u^{n+1} - u^n}{\Delta t} = Du^{n+1} + \frac{1}{\varepsilon^2}(1 - |u^n|^2)u^{n+1}, \tag{3.19}$$

where

$$D = \begin{pmatrix} T & I & & \\ I & T & I & \\ & I & T & I \\ & \ddots & \ddots & \ddots \\ & & I & T \end{pmatrix},$$

which is an m-by-m block diagonal matrix in which each block T and I is itself an m-by-m matrix,

$$T = \begin{pmatrix} -4 & 1 & & \\ 1 & -4 & 1 & \\ & 1 & -4 & 1 \\ & \ddots & \ddots & \ddots \\ & & 1 & -4 \end{pmatrix},$$

and I is an $m \times m$ identity matrix ([24]).

Therefore, we can write the equation (3.19) as,

$$\Big[I - \Delta t D - \frac{\Delta t}{\varepsilon^2}(I - |U^n|^2)\Big]U^{n+1} = U^n,$$

and in each step, we have to solve a linear system.

3.3.4 Convergence of Implicit method

Like for the explicit method in Section **3.2.3**, to show the convergence of the method, we will show the consistency and the stability of the method.

1. Consistency of the semi-implicit method
   Since the scheme is implicit, we expand the difference equation about the point (i, j, n+1) ([31]). Thus, we calculate the truncation error $\tau_{i,j}^{n+1}$ by using the Taylor series expansion for $u(x_i, y_j, t_{n+1})$. Let $u_{i,j}^n$ be the discretized version of $u(x_i, y_j, t_n)$. Our equation

$$u_t = \Delta u + f(u)$$

can be written in the finite difference scheme as:

$$\begin{aligned}\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t} = \frac{u_{i+1,j}^{n+1}-2u_{i,j}^{n+1}+u_{i-1,j}^{n+1}}{\Delta x^2} + \frac{u_{i,j+1}^{n+1}-2u_{i,j}^{n+1}+u_{i,j-1}^{n+1}}{\Delta y^2} \\ + f(u_{i,j}^{n}, u_{i,j}^{n+1}).\end{aligned} \tag{3.20}$$

Hence, the truncation error can be calculated as:

$$\begin{aligned}\tau_{i,j}^{n+1} := \frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t} - \frac{u_{i+1,j}^{n+1}-2u_{i,j}^{n+1}+u_{i-1,j}^{n+1}}{\Delta x^2} \\ - \frac{u_{i,j+1}^{n+1}-2u_{i,j}^{n+1}+u_{i,j-1}^{n+1}}{\Delta y^2} - f(u_{i,j}^{n}, u_{i,j}^{n+1}) - (u_t - \Delta u - f(u))\Big|_{x_i,y_j,t_n},\end{aligned}$$

using Taylor series expansion,

$$
\begin{aligned}
\tau_{i,j}^{n+1} :=& \frac{1}{\Delta t}\Big[u(x_i,y_j,t_{n+1}) \\
&- \Big(u(x_i,y_j,t_{n+1}) - \Delta t\, u_t(x_i,y_j,t_{n+1}) + \frac{\Delta t^2}{2}u_{tt}(x_i,y_j,t_{n+1}) + \cdots\Big)\Big] \\
&- \frac{1}{\Delta x^2}\Big[u(x_i,y_j,t_{n+1}) + \Delta x\, u_x(x_i,y_j,t_{n+1}) + \frac{\Delta x^2}{2}u_{xx}(x_i,y_j,t_{n+1}) + \\
&\qquad \frac{\Delta x^3}{6}u_{xxx}(x_i,y_j,t_{n+1}) + \frac{\Delta x^4}{24}u_{xxxx}(x_i,y_j,t_{n+1})\cdots \\
&\qquad - 2\,u(x_i,y_j,t_{n+1}) \\
&\qquad + u(x_i,y_j,t_{n+1}) - \Delta x\, u_x(x_i,y_j,t_{n+1}) + \frac{\Delta x^2}{2}u_{xx}(x_i,y_j,t_{n+1}) \\
&\qquad - \frac{\Delta x^3}{6}u_{xxx}(x_i,y_j,t_{n+1}) + \frac{\Delta x^4}{24}u_{xxxx}(x_i,y_j,t_{n+1})\cdots\Big] \\
&- \frac{1}{\Delta y^2}\Big[u(x_i,y_j,t_{n+1}) + \Delta y\, u_y(x_i,y_j,t_{n+1}) + \frac{\Delta y^2}{2}u_{yy}(x_i,y_j,t_{n+1}) \\
&\qquad + \frac{\Delta y^3}{6}u_{yyy}(x_i,y_j,t_{n+1}) + \frac{\Delta y^4}{24}u_{yyyy}(x_i,y_j,t_{n+1})\cdots \\
&\qquad - 2\,u(x_i,y_j,t_{n+1}) \\
&\qquad + u(x_i,y_j,t_{n+1}) - \Delta y\, u_y(x_i,y_j,t_{n+1}) + \frac{\Delta y^2}{2}u_{yy}(x_i,y_j,t_{n+1}) \\
&\qquad - \frac{\Delta y^3}{6}u_{yyy}(x_i,y_j,t_{n+1}) + \frac{\Delta y^4}{24}u_{yyyy}(x_i,y_j,t_{n+1})\cdots\Big] \\
&\qquad - f(u(x_i,y_j,t_n), u(x_i,y_j,t_{n+1})) \\
&- \left(u_t\,(x_i,y_j,t_{n+1}) - u_{xx}\,(x_i,y_j,t_{n+1}) - u_{yy}\,(x_i,y_j,t_{n+1}) - f(u(x_i,y_j,t_{n+1}))\right) \\
=& \frac{\Delta t}{2}u_{tt}(x_i,y_j,t_{n+1}) - \frac{\Delta x^2}{12}u_{xxxx}(x_i,y_j,t_{n+1}) - \frac{\Delta y^2}{12}u_{yyyy}(x_i,y_j,t_{n+1}) \\
&\qquad - f(u(x_i,y_j,t_n), u(x_i,y_j,t_{n+1})) + f(u(x_i,y_j,t_{n+1})).
\end{aligned} \tag{3.21}
$$

Since we know that as $\Delta t \to 0$, $f(u(x_i,y_j,t_{n+1})) \approx f(u(x_i,y_j,t_n))$, therefore,

$$
\begin{aligned}
|\tau_{i,j}^{n+1}| \le& \left|\frac{\Delta t}{2}u_{tt}(x_i,y_j,t_n)\right| + \left|\frac{\Delta x^2}{12}u_{xxxx}(x_i,y_j,t_n)\right| + \left|\frac{\Delta y^2}{12}u_{yyyy}(x_i,y_j,t_n)\right| \\
&+ |f(u(x_i,y_j,t_n)) - f(u(x_i,y_j,t_n), u(x_i,y_j,t_{n+1}))|
\end{aligned}
$$

or

$$
\lim_{\Delta x,\Delta y,\Delta t\to 0} |\tau_{i,j}^{n+1}| = 0.
$$

Hence, the scheme is consistent. If the scheme is fully implicit, in the last line of (3.21), the last term $f(u(x_i, y_j, t_n), u(x_i, y_j, t_{n+1})) = f(u(x_i, y_j, t_n))$, thus the scheme is consistent.

2. Stability of the implicit method

   In order to show the stability, we again use the discrete von Neumann criterion for stability ([31]). To apply the discrete von Neumann criterion to a two dimensional problem, we consider a discrete Fourier mode for the problem of the form

$$u^n_{j,k} = \xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y} \tag{3.8}$$

where $i^2 = -1$ and p, q are the wave numbers for x and y respectively.

As we explained in the stability analysis of the explicit method in Section **3.2.3**, if ε is large enough, we can ignore the nonlinear term in (3.1). A similar calculation for the implicit method shows that ε has to satisfy the condition of $\varepsilon \geq h\sqrt{\frac{1}{8}}$. Hence, for the stability analysis, we ignore the nonlinear term in (3.1).

If we plug the expression into the implicit equation (excluding the last term $f(u^n_{i,j})$),

$$\frac{u^{n+1}_{i,j} - u^n_{i,j}}{\Delta t} = \frac{u^{n+1}_{i+1,j} - 2u^{n+1}_{i,j} + u^{n+1}_{i-1,j}}{\Delta x^2} + \frac{u^{n+1}_{i,j+1} - 2u^{n+1}_{i,j} + u^{n+1}_{i,j-1}}{\Delta y^2} \tag{3.22}$$

becomes

$$\begin{aligned}
&\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} = \xi^n e^{ijp\pi\Delta x + ikq\pi\Delta y} \\
&+ \frac{\Delta t}{\Delta x^2}[\xi^{n+1} e^{i(j+1)p\pi\Delta x + ikq\pi\Delta y} - 2\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} + \xi^{n+1} e^{i(j-1)p\pi\Delta x + ikq\pi\Delta y}] \\
&+ \frac{\Delta t}{\Delta y^2}[\xi^{n+1} e^{ijp\pi\Delta x + i(k+1)q\pi\Delta y} - 2\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} + \xi^{n+1} e^{ijp\pi\Delta x + i(k-1)q\pi\Delta y}].
\end{aligned} \tag{3.23}$$

With the trigonometric identities $\cos(z) = \frac{e^{iz}+e^{-iz}}{2}$ and $\cos(2x) = 1 - 2\sin^2(x)$, we simplify the equation (3.23), and we get,

$$\begin{aligned}
&\xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y} = \xi^{n} e^{ijp\pi\Delta x + ikq\pi\Delta y} \\
&+ \xi^{n+1} e^{ijp\pi\Delta x + ikq\pi\Delta y}\Big[\frac{\Delta t}{\Delta x^2}(e^{ip\pi\Delta x} - 2 + e^{-ip\pi\Delta x}) + \frac{\Delta t}{\Delta y^2}(e^{iq\pi\Delta y} - 2 + e^{-iq\pi\Delta y})\Big].
\end{aligned}$$

Hence,

$$\begin{aligned}
\xi &= 1 + \xi\Big[\frac{\Delta t}{\Delta x^2}(2\cos(p\pi\Delta x) - 2) + \frac{\Delta t}{\Delta y^2}(2\cos(q\pi\Delta y) - 2)\Big] \\
&= 1 + \xi\Big[\ 2\frac{\Delta t}{\Delta x^2}(\cos(p\pi\Delta x) - 1) + 2\frac{\Delta t}{\Delta y^2}(\cos(q\pi\Delta y) - 1)\Big] \\
&= 1 + \xi\Big\{\ 2\frac{\Delta t}{\Delta x^2}\Big[1 - 2\sin^2(\frac{p\pi\Delta x}{2}) - 1\Big] + 2\frac{\Delta t}{\Delta y^2}\Big[1 - 2\sin^2(\frac{q\pi\Delta y}{2}) - 1\Big]\Big\} \\
&= 1 - \xi\Big[\ 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{p\pi\Delta x}{2}) + 4\frac{\Delta t}{\Delta y^2}\sin^2(\frac{q\pi\Delta y}{2})\Big].
\end{aligned}$$

If we solve for ξ, we get

$$\begin{aligned}
&\xi\Big[1 + 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{p\pi\Delta x}{2}) + 4\frac{\Delta t}{\Delta y^2}\sin^2(\frac{q\pi\Delta y}{2})\Big] = 1. \\
\Rightarrow\quad &\xi = \frac{1}{1 + 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{p\pi\Delta x}{2}) + 4\frac{\Delta t}{\Delta y^2}\sin^2(\frac{q\pi\Delta y}{2})}. \qquad (3.24)
\end{aligned}$$

Since $0 \leq \sin^2(x) \leq 1$ for any argument x, we have always $|\xi| \leq 1$.

Therefore, the scheme is unconditionally stable provided ε is in the allowed range.

Chapter 4

Analysis of Results

4.1 Colour Images

In this section, we focus on colour images. Mathematically, a colour image can be represented as a mapping

$$u : \Omega \to \mathbb{C}^3.$$

(Each of the three components refer to R(ed), G(reen), and B(lue), respectively.)

4.1.1 Using the max norm

[20] says that a common approach to inpaint colour images is to inpaint the colour component separately, but since in real world images, the colour components are typically not independent, a separation approach may lead to artifacts, like spurious colours or rainbow effects. Therefore, if u is a colour image, it is better for the equation

$$\Delta u + \frac{1}{\varepsilon^2}(1 - |u|^2)u = 0 \tag{2.6}$$

to replace the Euclidian distance by an appropriate norm in $\mathbb{C}^n$. For RGB colour images, the maximum norm of the RGB-components is most appropriate. It is defined as:

$$||u(x)|| := \max\{|u^1(x)|, |u^2(x)|, |u^3(x)|\}.$$

The goal of this section is to show the difference in results when each component of the colour image is processed separately and when the maximum component of the RGB colour image is used as the norm instead.

4.1.2 Hypothesis for the Spurious Colour

When we take the maximum of colour components and replace $|u|$ in (2.6) with $||u(x)|| := \max\{|u(\mathrm{R})|, |u(\mathrm{G})|, |u(\mathrm{B})|\}$, then the value of $||u||$ in each component has the same value. Therefore, $||u||$ is always a grey scale image, and the largest value corresponds to white (1), and the lowest value to black (0). Thus, $1 - ||u||$ is also the grey scale image, and as a result, it will not cause spurious colours.

On the other hand, when each colour is processed separately, the largest value of each colour remains as it is. And, most likely, the highest or lowest value of each colour occurs on a different pixel in different colour component, they will not cancel each other. Therefore, the highest values are added during the iterations, and it might result in spurious colour or rainbow effect.

Based on the hypothesis above, the following experiment was carried out.

4.1.3 Experimental result for the Hypothesis

For this experiment, we create a 10-by-10 colour image u. The image u has three components R, B and G so that each component is a 10-by-10 matrix with random numbers between 0 and 255. Then, we scale u to the complex value so that the same value in each of three colour components of u in the range of $[0, 255]$ is mapped to the same value in the range of $[-1, 1]$ to get the real part of $\mathcal{R}(u)$. And the imaginary part is calculated as $\mathcal{I}(u) = \sqrt{1 - \mathcal{R}(u)^2}$.

Figure 4.1 is a 10-by-10 matrix created by random numbers between 0 and 255. Then, we scale the image into the complex value as explained above. Only the real part is used to obtain the image. The left image in Figure 4.2 is the image of $||u|| = \max\{|u(R)|, |u(G)|, |u(B)|\}$, and the right image in Figure 4.2 is the image of $|u|$. For the image of $||u||$, since the maximum value among three components is used, all three components have the same value. Therefore, the image which is the combination of three colour components is grey scaled. On the other hand, for the image of $|u|$, since each colour component has different value, it is a colour image.

From this experiment, at least we notice that the image of $|u|$ tends to have more colours than the image of $||u||$. Also, we have tried many other experiments to see the differences in these two methods. But, we could not find any clearer differences than this result. Now, the question is how strongly image inpainting of a real picture is affected by this strategy.

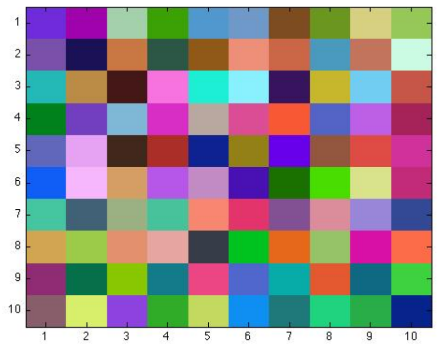

Figure 4.1: Original Image

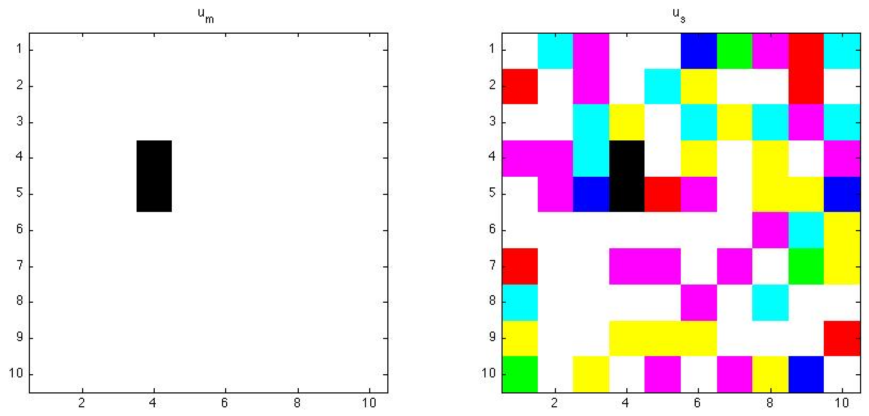

Figure 4.2: Left: $||u|| = \max\{|u(R)|, |u(G)|, |u(B)|\}$, Right: $|u|$

4.1.4 Experiment with a Real Picture

We repeat the experiment with a real picture. The original parrot image has the fence in front of the parrot. We will use the Ginzburg-Landau equation to delete the fence. The mask image is used to define the inpainting domain Ω. The process was iterated 500 times with $\varepsilon = 0.7$ and the time step $dt = 0.1$.

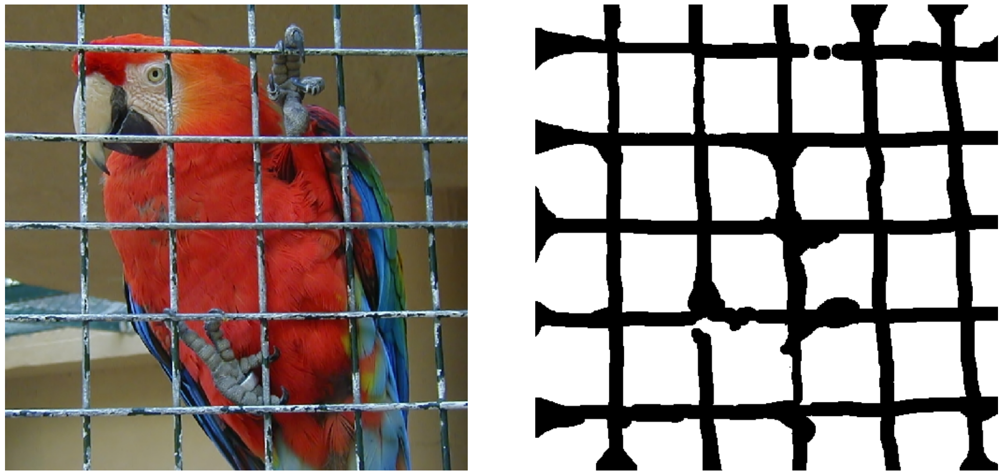
Figure 4.3: Original image (left) and mask image (right)

Figure 4.4 shows the inpainted images when the maximum value is used for $|u|$ (the top row) and when colour components are processed separately, and combined after the process (the bottom row).

We can hardly see a difference in these two images. But when we look at plots of colour values in the right column of Figure 4.4, we notice that the final values of blue and green are higher in the separated case than the ones when the maximum value was used. And when we look at Table 4.1, we confirm that even though the final values of three colours in both methods are very similar, the values of blue and green in the separated method are higher than when the maximum value is used.

Therefore, we conclude that the inpainted image produced by using the method of assigning the maximum value of three colours is not very different from the one produced by using the method of separating components first and combining later. But, when we

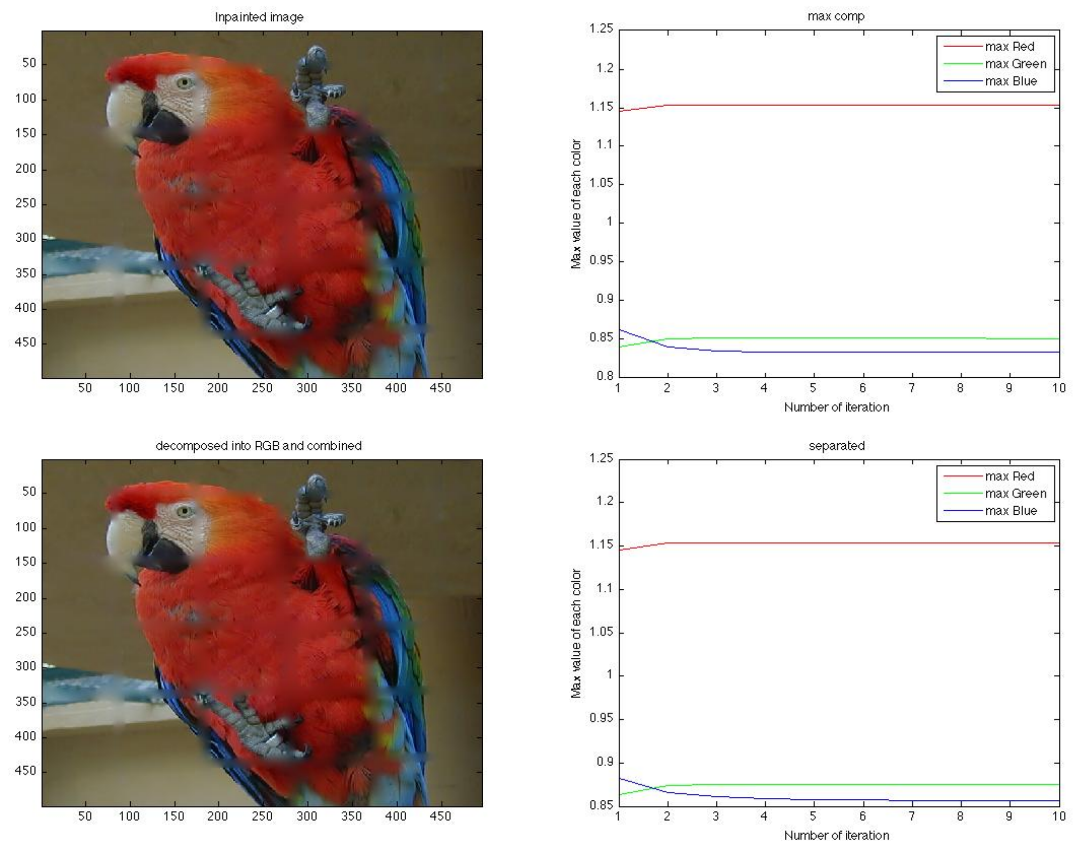

Figure 4.4: Inpainted image and Colour Distribution
Top: The max value is used
Bottom: Processed separately

	Max Value			Separated		
Iteration	R	G	B	R	G	B
50	1.1457	0.8406	0.8630	1.1457	0.8633	0.8826
100	1.1531	0.8507	0.8406	1.1531	0.8744	0.8645
150	1.1535	0.8514	0.8343	1.1535	0.8752	0.8606
200	1.1536	0.8514	0.8340	1.1536	0.8752	0.8588
250	1.1536	0.8514	0.8340	1.1536	0.8752	0.8579
300	1.1536	0.8514	0.8340	1.1536	0.8752	0.8574
350	1.1535	0.8514	0.8340	1.1535	0.8752	0.8570
400	1.1535	0.8513	0.8340	1.1535	0.8751	0.8567
450	1.1535	0.8513	0.8340	1.1535	0.8751	0.8566
500	1.1535	0.8513	0.8340	1.1535	0.8751	0.8564

Table 4.1: Maximum values List for Colour Distribution of the whole image

check the value of each colour in the inpainted image, we notice that when the colours are processed separately, we tend to have higher values of colour components.

4.2 Size and Shape of the Inpainting Area

In this section, we investigate how the size of the mask affects the region to be inpainted. We let the region which is defined by a mask be Ω, and the missing part of the original image be Γ here. All experiments in this section are done with $\varepsilon = 0.1$, $\Delta t = \frac{\varepsilon^2}{4}$ if not stated otherwise, and the algorithm was iterated until the indicated numbers.

If we use the complex-valued Ginzburg-Landau equation with 1000 iterations to inpaint the image with a vertical white bar of the width 50 pixels, and a mask of size 150-by-300 pixels, we get the result in Figure 4.5.

As we can see from the inpainted image, the complex-valued Ginzburg-Landau equation does not minimize the length of the missing region's edges. Since the region to be filled-in (Γ) has 150 pixels as the vertical length and 50 pixels as the horizontal length, if the equation minimizes the length of the edges of Γ, the white vertical bar should be cut into two parts.

Here, we will investigate how the Ginzburg-Landau equation fills in the missing part.

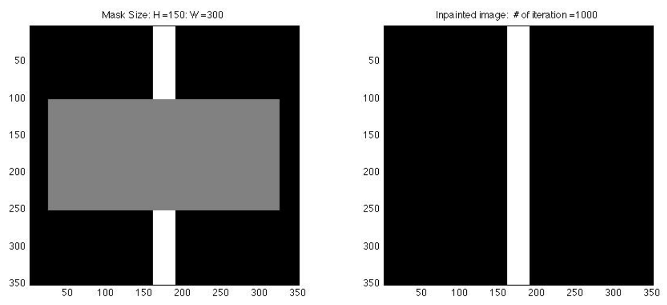

Figure 4.5: Left: Original Image (Ω:150-by-300, Line width: 50) Right: Inpainted image

4.2.1 Inpainting the missing region Γ when Γ is a white bar

Now, we check how Ω is filled in during the iteration. We use the image of size 350-by-350 pixels.

Case 1 : The image with a vertical white bar

The following two examples are done for two white vertical bars of width 30 pixels and 50 pixels, with Ω has height 250 and width 300 pixels, and height 310 and width 100 pixels, respectively. The algorithm was iterated for each case 1600 times and 700 times, respectively.

As we can see in Figure 4.6 and 4.7, Ω is shrinking towards the centre of the square by the same length vertically and horizontally. All blue, pink, green and yellow lines that enclose the shrinking Ω are equally spaced. This is because the complex Ginzburg-Landau equation is searching for the stable point in the complex unit circle, $\{x \in \mathbb{C} : |x| = 1\}$. Ω shrinks by the same amount (i.e. the same number of pixels) vertically and horizontally.

Since Ω shrinks by the same amount horizontally and vertically, we are concerned about the square region located in the centre of the image. Let these squares of 30-by-30 pixels and 50-by-50 pixels in the centre of the white bar be Λ. The differences in the inpainted images of Figure 4.6 and 4.7 are the size of Ω. If the size of Ω satisfies the following condition, the vertical bar is recovered.

$$(\textbf{Height of } \Omega) < (\textbf{Width of } \Omega) - (\text{Line width})$$

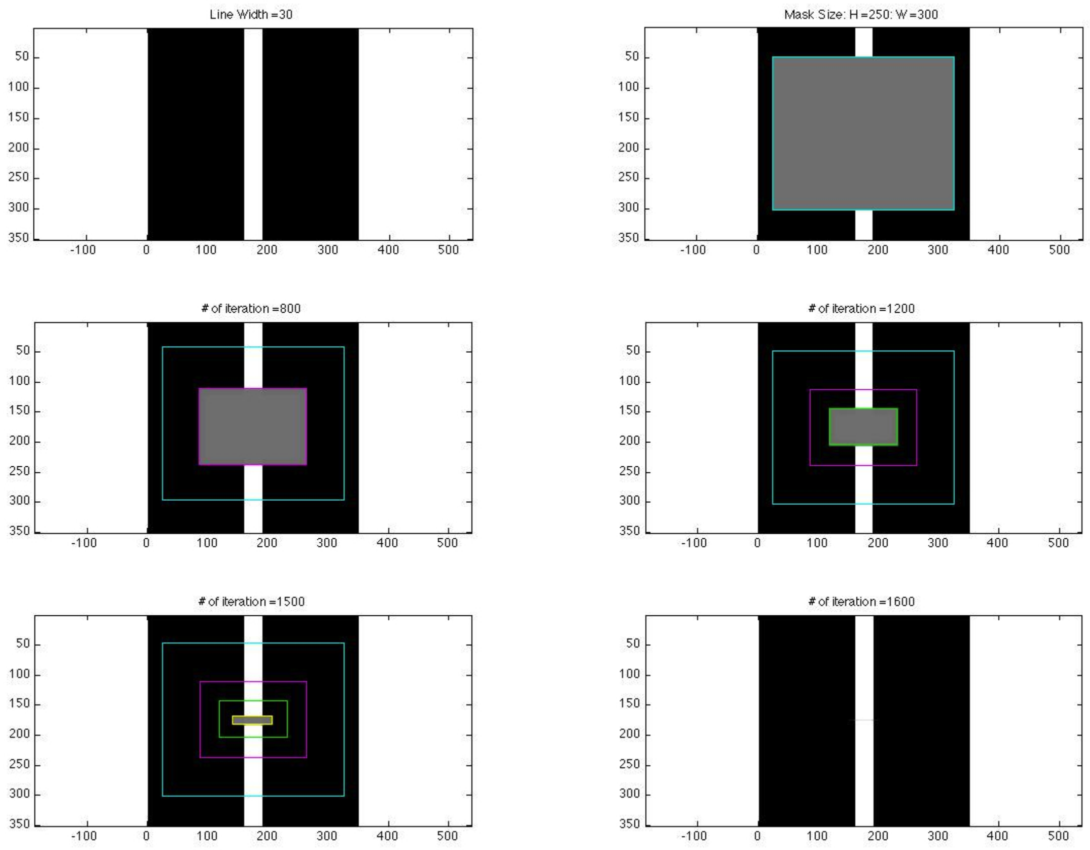

Figure 4.6: Line width=30, Mask size: height = 250, width = 300, after 1600 iterations

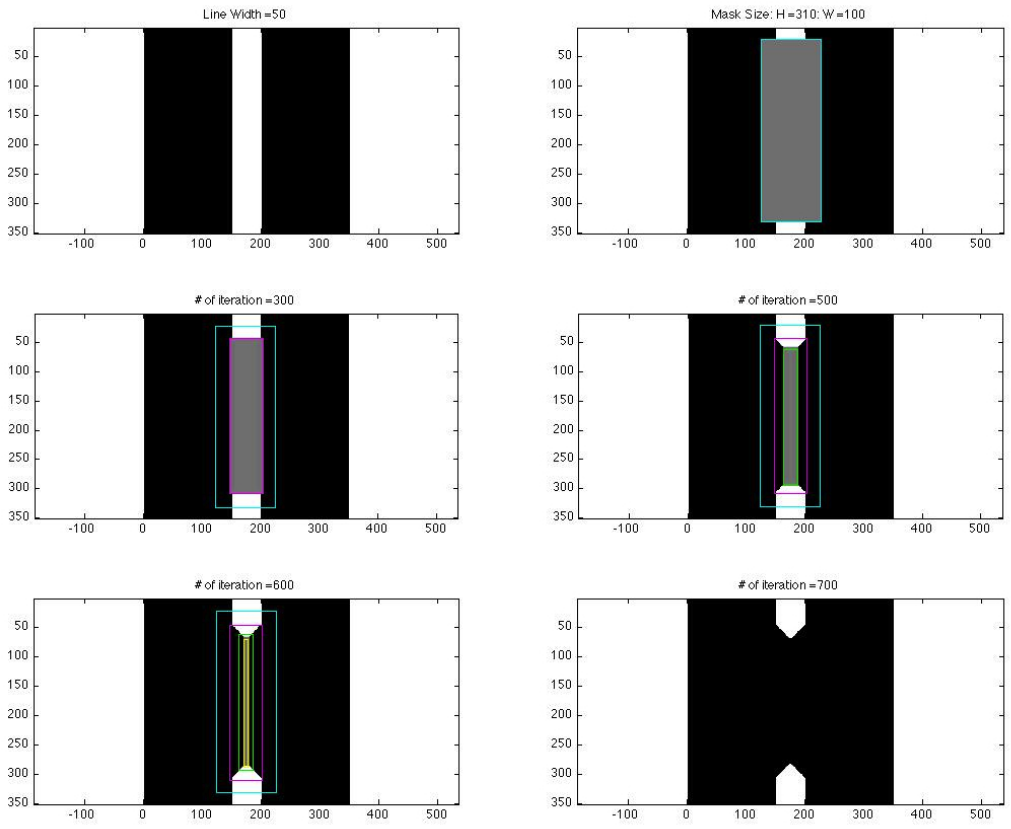

Figure 4.7: Line width=50, Mask size: height = 310, width = 100, after 700 iterations

As Ω shrinks by the same amount vertically and horizontally, if the vertical edges of Ω touch Λ before the horizontal edges of Ω are being pushed down and pushed up towards the centre of Λ, the vertical edges shrink towards Λ, and the result is Figure 4.7. If the horizontal edges of Ω are pushed down from the top and pushed up from the bottom to the centre of Λ before the vertical edges of Ω touch Λ, we recover the vertical white bar.

Case 2 : The image with a horizontal white bar

The symmetric argument holds for the image with a horizontal white bar. If the size of Ω satisfies the following condition, the horizontal bar is recovered.

$$(\textbf{Width of } \Omega) < (\textbf{Hight of } \Omega) - (\text{Line width})$$

As we can see in Figure 4.8 and 4.9 again, Ω is shrinking towards the centre of the square by the same length vertically and horizontally. The experiments are done with $\varepsilon = 0.5$ and $\Delta t = 0.1$. We notice that the bigger time step requires fewer number of iterations.

Here, we have checked the case when the object to be recovered was a line. If the region to be recovered (Γ) is a square not a line, then as long as the mask covers Γ, the original image is recovered.

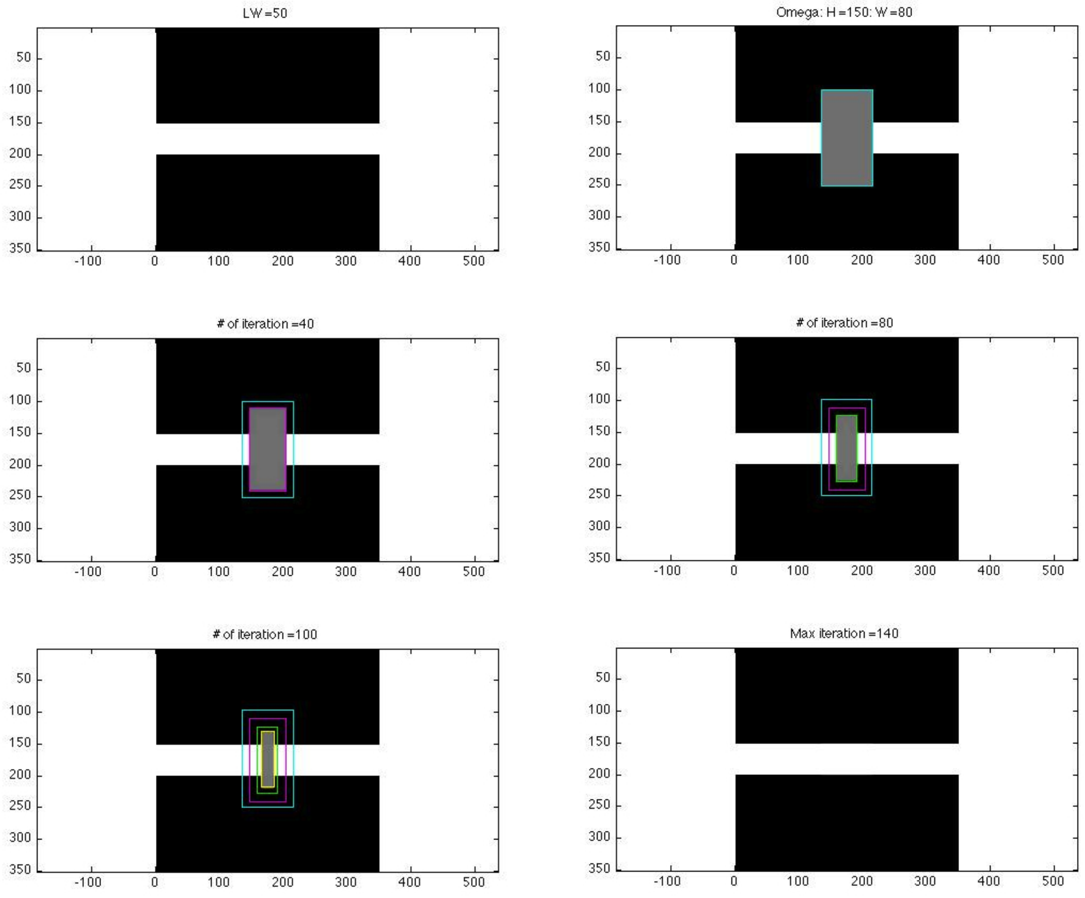

Figure 4.8: Line width=50, Mask size: height=150, width=80, after 140 iterations

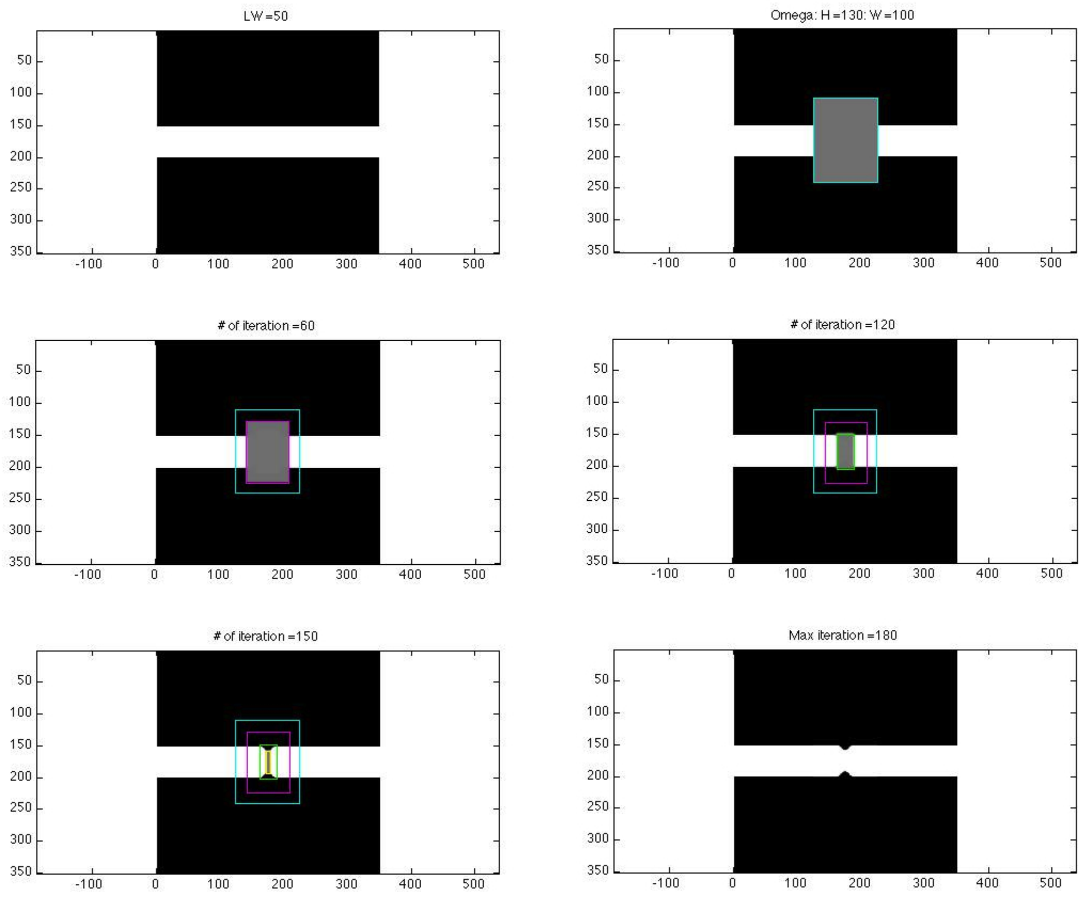

Figure 4.9: Line width=50, Mask size: height=130, width=100, after 180 iterations

4.2.2 Inpainting the missing region Γ when Γ has several levels of brightness

Next, we investigate when the region to be inpainted has more than one level of brightness. The original image has three levels of brightness, dark grey, bright grey and white. The image is the size of 350-by-350 pixels. We divide it into 7 bars where each bar has a width of 50 pixels (i.e. each bar has the same width). Therefore, in this case, the line width is 250 pixels.

The following two examples are done for this image with Ω which has height 48 and width 300 pixels, and height 100 and width 300 pixels, respectively. Experimentally, we have found that the energy of the image with Ω which has height 48 and width 300 becomes negative after 332 iterations. Therefore, we investigate the inpainted image after 332 iterations, and the inpainted image after further iterations, in this experiment, we choose 600 iterations.

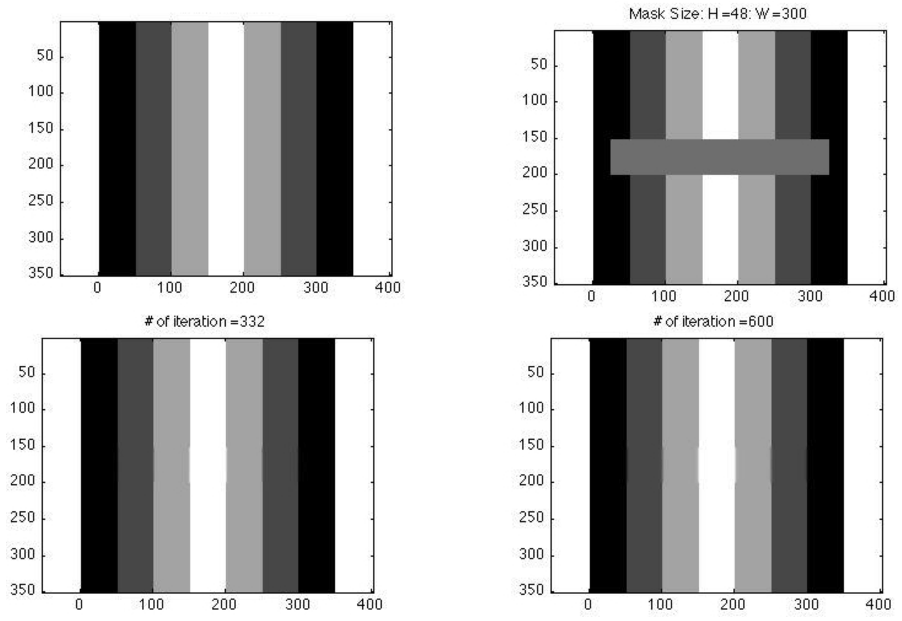

Figure 4.10: **Top** Left: Original Image with Line width=250, Right: Mask with height=48, width=300, **Bottom** Inpainted Image Left: after 332 iterations, Right: after 600 iterations

In Figure 4.10, the length of the height (48) of Ω is less than the length of the width (300) minus the line width (250). Therefore, the original image is recovered after 332 iterations. (As we mentioned before, after this number, the energy becomes negative.) We notice that if we iterate further until 600 iterations, the output image does not change from the output image with 332 iterations.

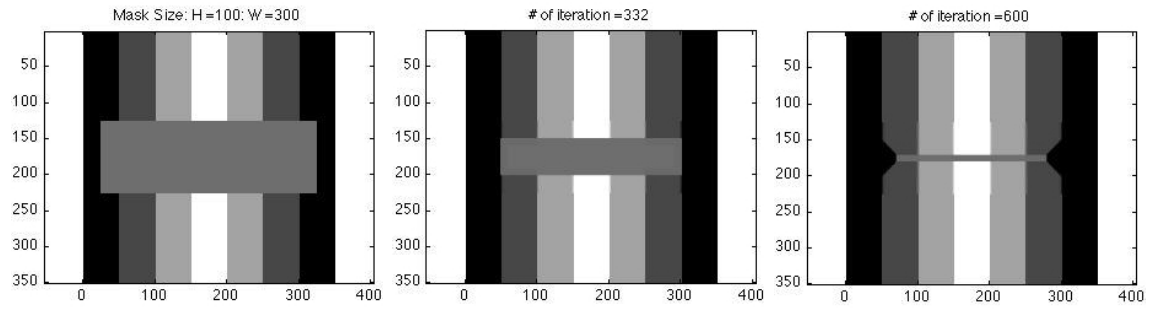

Figure 4.11: Left: Mask with height=100, width=300
Middle, Right: Inpainted Images after 332 and 600 iterations respectively

On the other hand, in Figure 4.11, the length of the height (100) of Ω is greater than the length of the width (300) minus the line width (250). Therefore, as soon as the vertical edges of Ω touch the edge of the darker grey bars (this happens around 332 iterations) before the horizontal edges of Ω being pushed down and pushed up 50 pixels towards the centre of the image, the vertical edges of Ω shrink towards the centre. The result is not what we expect for the inpainting after 600 iterations.

We can see the same phenomenon in Figure 4.12. This example shows when the mask is vertically longer. The vertical edges of Ω touche the white bar around 322 iterations, and after this, the bright grey part shrinks towards the centre.

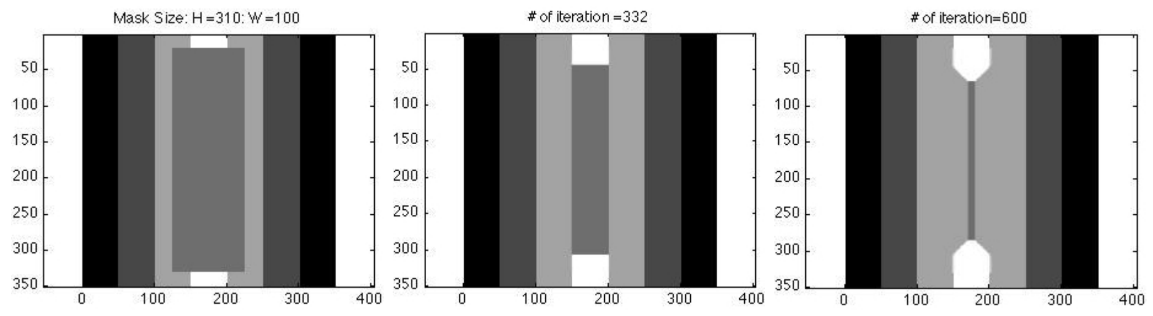

Figure 4.12: Left: Mask with height=310, width=100
Middle, Right: Inpainted Images after 332 and 600 iterations respectively

4.2.3 Inpainting the missing region Γ when Γ is a part of the circle

In this section, we investigate the image with a circle. In this case, the region to be inpainted, Ω is a square.

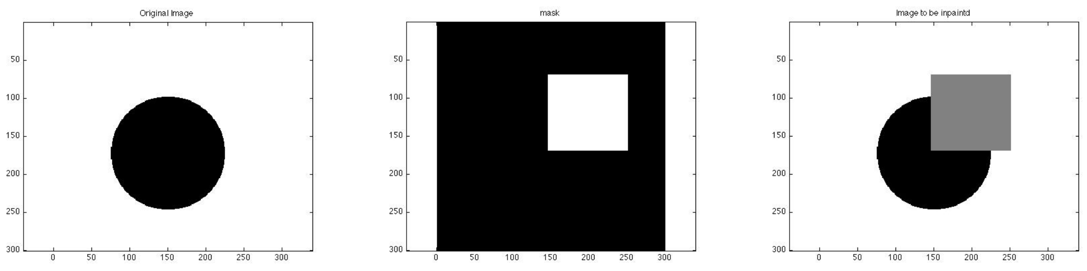

Figure 4.13: Left: The original Image, Middle: The mask Image, Right: The image to be inpainted

The results and the energy plot are shown in Figure 4.14. We notice that after 200 iterations, the edge of the circle is inpainted with the straight line which is the diagonal of the square in the left top image. From the energy plot, we notice that the energy value does not change very much after 200 iterations. Hence, it converges after 200 iterations. If we iterate further, the image gets blurred.

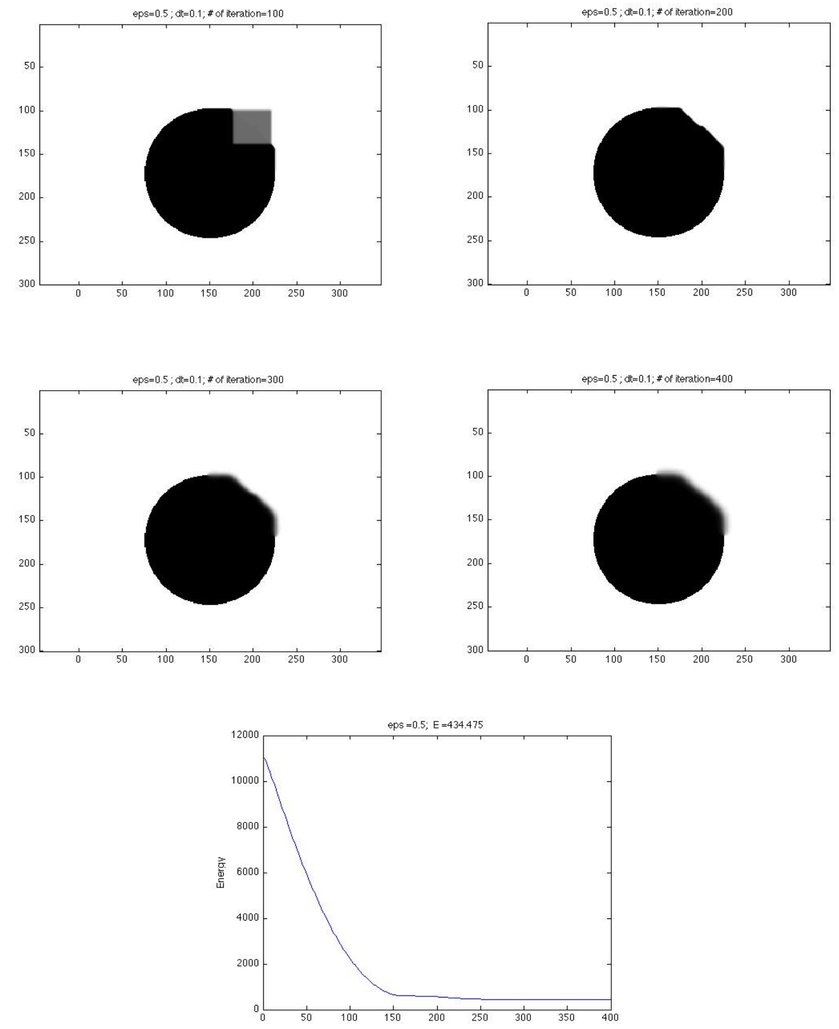

Figure 4.14: Inpainted images and Energy plot
Top Left: after 100 iterations, Right: after 200 iterations
Middle Left: after 300 iterations, Right: after 400 iterations
Bottom: The energy plot

4.3 The relationship between the value of ε and the time step size Δt

In [20], if the image is ambiguous (the left image in Figure 4.15), the noisy area should be inpainted as the right image in Figure 4.15, and if the level set algorithm is used, the inpainted image would be the ones in Figure 4.16.

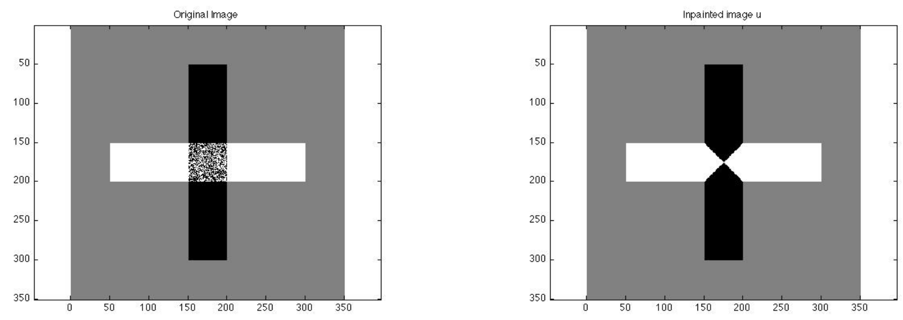

Figure 4.15: Left : Ambiguous image, Right: via Ginzburg-Landau algorithm **u**

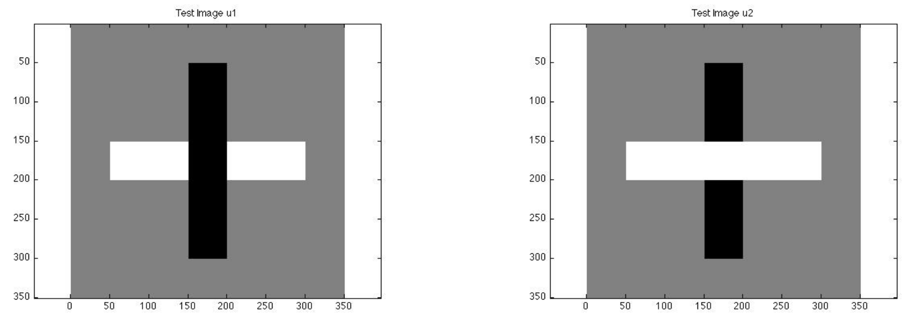

Figure 4.16: via Level set algorithm $u1$, $u2$

The inpainted image of this ambiguous image describes the peculiarity of the Ginzburg-Landau equation. Therefore, in this section, the dependence of the Ginzburg-Landau equation and the value of ε, and the relationship between the value of ε and the time step size Δt are investigated experimentally for the explicit and implicit methods using this ambiguous image in Figure 4.15.

4.3.1 The results for the Explicit method

For the explicit method, various values of time step size Δt, ε and number of iterations are used. We use the spatial step size $\Delta x = 1$ and $\Delta y = 1$.

- time step size: $\Delta t = \{0.01, 0.05, 0.1, 0.2\}$.
  Note: The time step size has to be $\frac{\Delta t}{\Delta x^2} + \frac{\Delta t}{\Delta y^2} \leq \frac{1}{2}$ from the stability analysis.
  Since $\Delta x = 1$ and $\Delta y = 1$, we need $\Delta t \leq \frac{1}{4}$.

- the values of $\varepsilon = \{0.01, 0.1, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$.
  Note: ε has to satisfy $\varepsilon \geq h\sqrt{\frac{1}{8}}$ if we let $\Delta x = \Delta y = h$ from the stability analysis.
  Since $\Delta x = 1$ and $\Delta y = 1$, we need $\varepsilon \geq \sqrt{\frac{1}{8}} \approx 0.36$.

- the number of iterations = $\{100, 500, 1000\}$

- The energy is plotted for $\Delta t = \{0.05, 0.1\}$ with the number of iterations = 500.

Results:

1. Examples of the inpainted image

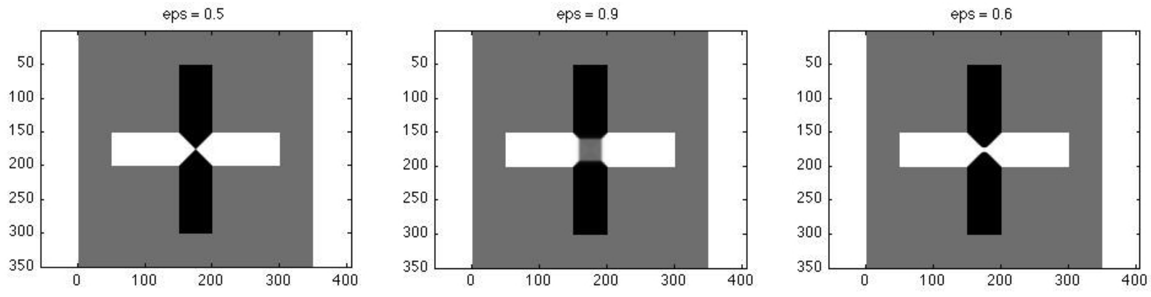

Figure 4.17: Inpainted Images
Left ◯ : Expected results,
Middle ■ : grey square at the centre,
Right □ : space in the vertical bar

Note that Symbols ◯, ■ and □ are used in Table 4.2.

2. Energy plot

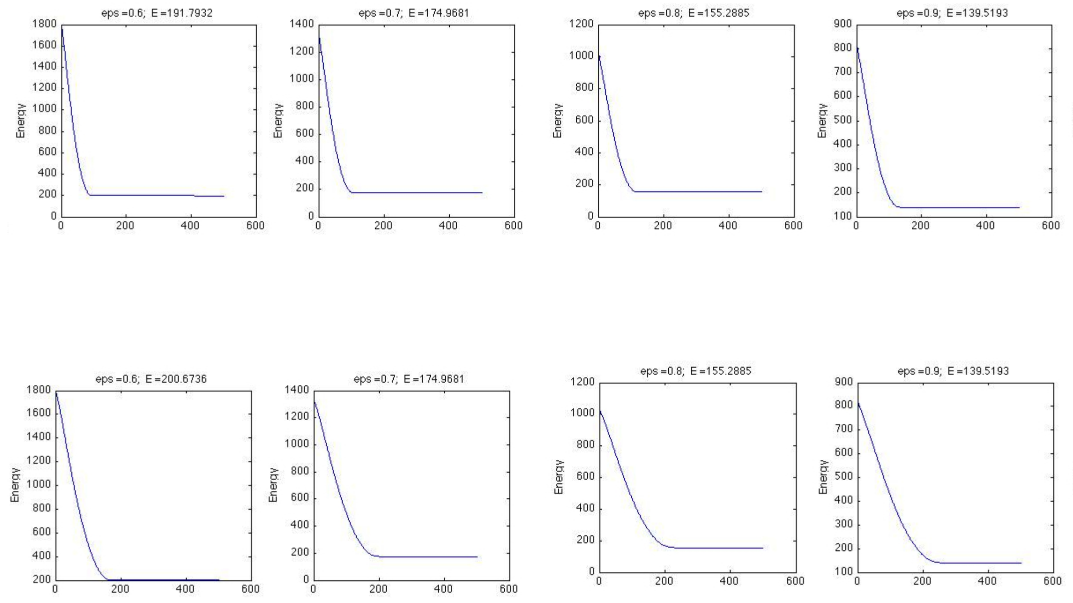

Figure 4.18: Energy Plots
Row 1: Iteration = 500, $\Delta t = 0.1$, $\varepsilon = \{0.6, 0.7, 0.8, 0.9\}$
Row 2: Iteration = 500, $\Delta t = 0.05$, $\varepsilon = \{0.6, 0.7, 0.8, 0.9\}$

Summary:

- In general, for smaller ε, it is better to use smaller Δt, and if ε is closer to 1, Δt can be close to the stability restriction $\frac{1}{4}$.
- When we look at the energy plots for $\Delta t = \{0.05, 0.1\}$ in Figure 4.18, the final values of the total energy are exactly the same for $\varepsilon = \{0.7, 0.8, 0.9\}$ as well as the output images look very similar. In this case, the final energy of the inpainted image depends on the values of ε not on the value of the time step Δt.

	values of ε / # of iterations	0.01	0.1	0.3	0.5	0.6	0.7	0.8	0.9	1
$\Delta t = 0.01$	100	╱	○ at 0.09	■	■	╱	■	╱	╱	-
	500	$\times$	$\otimes$	○	■	■	■	■	■	■
	1000	$\times$	$\otimes$	□	○	○	○	■	■	■
$\Delta t = 0.05$	100	$\times$	$\times$	○	■	■	■	■	■	■
	500	$\times$	$\times$	○	○	○	○	○	○	○
	1000	$\times$	$\times$	○	□	□	○	○	○	○
$\Delta t = 0.1$	100	$\times$	$\times$	$\otimes$	○	○	○	■	■	■
	500	$\times$	$\times$	$\otimes$	□	□	○	○	○	○
	1000	$\times$	$\times$	$\otimes$	□	□	★	★	○	○
$\Delta t = 0.2$	100	$\times$	$\times$	$\times$	$\times$	$\otimes$	$\otimes$	○	○	○
	500	$\times$	$\times$	$\times$	$\times$	$\otimes$	$\otimes$	$\otimes$	$\otimes$	○
	1000	$\times$	$\times$	$\times$	$\times$	$\otimes$	$\otimes$	$\otimes$	$\otimes$	○

Notation on the table:

$\times$:= black vertical bar,	$\otimes$:= noise,
○ := expected result,	■ := grey square at the centre,
□ := space in the vertical bar,	╱ := not checked,
$-$: = no change,	★ := vertical black bar is connected

Table 4.2: Results of 4.4.1

4.3.2 The results for the Semi-Implicit method

For the semi-implicit method, various values of time step size Δt, ε and number of iterations are used. We use the spatial step size $\Delta x = 1$ and $\Delta y = 1$.

- time step size: $\Delta t = \{0.01, 0.05, 0.1, 0.3, 0.5, 0.9\}$.
  Note: There is no restriction for the time step size for the implicit method, but ε has to satisfy the condition below.
- the values of $\varepsilon = \{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$.
  Note: ε has to satisfy the condition of $\varepsilon \geq h\sqrt{\frac{1}{8}}$ if we let $\Delta x = \Delta y = h$ from the stability analysis. Since $\Delta x = \Delta y = 1$, $\varepsilon \geq \sqrt{\frac{1}{8}} \approx 0.36$.
- the number of iterations $= \{50, 100\}$
- The energy is plotted for $\Delta t = \{0.05, 0.1\}$ with the number of iterations $= 100$.

We notice that we can use a wider range of Δt with the implicit method than with the explicit method. Also, we notice that the number of iterations required are fewer in the implicit method.

Results:

1. Examples of the inpainted image
   For the semi-implicit method, the expected result is that the vertical bar is connected as the left image of Figure 4.19. In out experiment, we could not get any images that had disconnected vertical black bars with this method.
2. Energy plots
   With $\varepsilon = \{0.7, 0.9\}$ with $\Delta t = 0.05$, the inpainted images have grey squares at the centre. As we can see from the energy plot in Figure 4.20, the energy has not reached a stable condition yet.

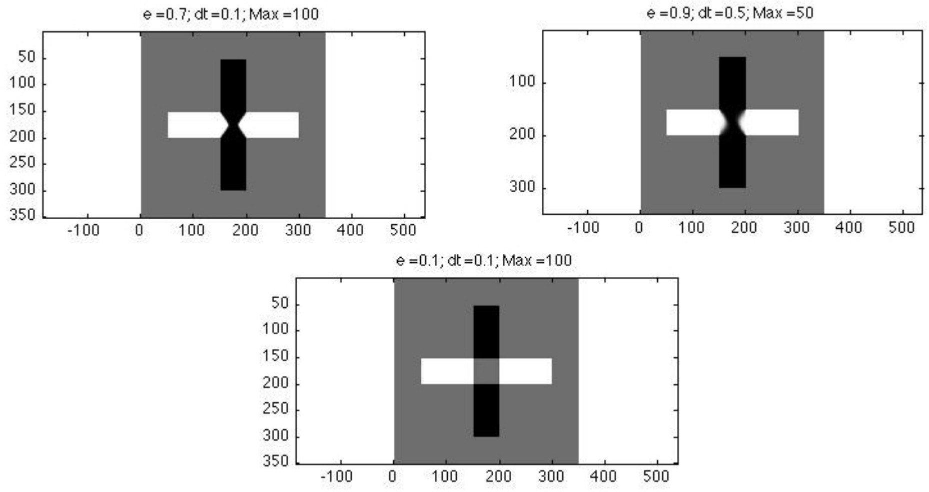

Figure 4.19: Examples of the inpainted image with the semi-implicit method
Top: Left $\bigcirc$: Expected result, Right $\circ$: Good result
Bottom: $-$: No change from the beginning of the iteration

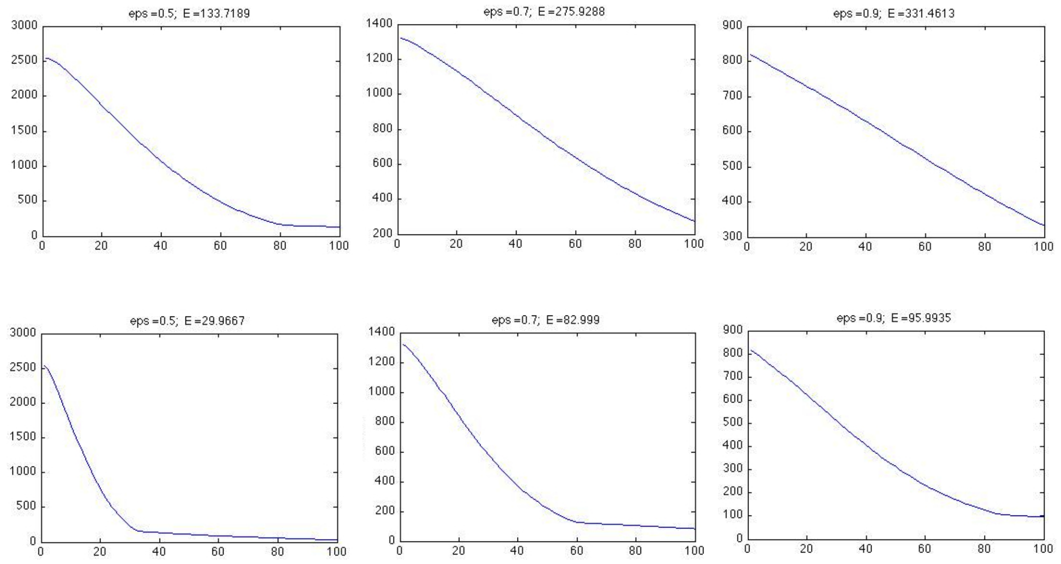

Figure 4.20: Energy Plots for the semi-implicit method
Row 1: Iteration = 100, $\Delta t = 0.05$, $\varepsilon = \{0.5, 0.7, 0.9\}$
Row 2: Iteration = 100, $\Delta t = 0.1$, $\varepsilon = \{0.5, 0.7, 0.9\}$

Summary:

- When we compare the results of the semi-implicit method and the explicit method, we can see in the results in Table 4.3 that, with the semi-implicit method, we can use a wider range of time step Δt, in particular we can use larger Δt. This is the advantage of a implicit method. The other advantage of the implicit method is that the number of iterations required is fewer than for the explicit method.

- The disadvantage of the semi-implicit method is that it takes a long time to run the program. Since we are dealing with an image array not a vector, in order to solve the linear system for U^{n+1}, we have to convert the image into a vector. As the size of the image gets larger, the size of the vector gets much larger. Here, we use the image of size 350-by-350. Hence the corresponding vector size is 350^2-by-1. It means that the discretization matrix for the Laplacian is 350^2-by-350^2 which is a large sparse matrix. Therefore, it takes time to calculate the value of U^{n+1} in matlab.

- Also, when we look at the results in Table 4.2 and 4.3, we notice that in order to get the expected results with the semi-implicit method, it is better to use Δt in the range of 0.05, 0.1 or maybe 0.3. Since these values can be also used with the explicit method, if we think about the computation time, even though the number of iterations is fewer, there is no significant advantage to choosing the semi-implicit method over the explicit method.

	values of ε / # of iterations	0.1	0.3	0.5	0.7	0.9	1
$\Delta t = 0.01$	50	G	■	■	■	■	■
	100	G	■	■	■	■	■
$\Delta t = 0.05$	50	-	◯	■	■	■	■
	100	-	◯	◯	■	■	■
$\Delta t = 0.1$	50	-	$\otimes$	◯	■	■	■
	100	-	-	◯	◯	◯	◯
$\Delta t = 0.3$	50	-	G	$\otimes$	$\circ$	◯	◯
	100	-	B	-	$\circ$	$\circ$	$\circ$
$\Delta t = 0.5$	50	-	G	$\otimes$	$\otimes$	$\circ$	$\circ$
	100	-	G	-	$\otimes$	$\circ$	$\circ$
$\Delta t = 0.9$	50	-	-	G	-	$\otimes$	$\circ$
	100	-	-	-	-	$\otimes$	$\circ$

Notation on the table:

◯:= expected result,	$\circ$:= good result,
■:= grey square at the centre,	$\otimes$:= noise,
G := grey image,	B := black image,
$-$: = no change	

Table 4.3: Results of 4.4.2

4.3.3 The results for the Fully Implicit method

For the fully implicit method, various values of time step size Δt, ε and number of iterations are used. We use the spatial step size $\Delta x = 1$ and $\Delta y = 1$.

- time step size: $\Delta t = \{0.05, 0.1, 0.2, 0.3, 0.5\}$.
  Note: There is no restriction for the time step size for the implicit method, but ε has to satisfy the condition below.
- the values of $\varepsilon = \{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$.
  Note: ε has to satisfy the condition of $\varepsilon \geq h\sqrt{\frac{1}{8}}$ where $\Delta x = \Delta y = h$ from the stability analysis. Since $\Delta x = \Delta y = 1$, $\varepsilon \geq \sqrt{\frac{1}{8}} \approx 0.36$
- the number of iterations $= \{50, 100\}$
- The energy is plotted for $\Delta t = \{0.2, 0.3\}$ with the number of iterations $= 50$.

We notice that we can use a wider range of Δt with the fully implicit method, the same as with the semi-implicit method. Also, we notice that the number of iterations required is fewer for the fully implicit method than for the explicit method.

As we mentioned in Chapter 3, the fully implicit method requires the discrete Laplacian matrix, an identity matrix and a Jacobian matrix of the size m^2-by-m^2. Here, we use the image of size 350-by-350. As we use 350^2-by-350^2 matrices, it takes a long time for the computation.

Results:

1. Examples of the inpainted image

   The same as the semi-implicit method, for the fully implicit method, the expected result is that the vertical bar is connected. In our experiment, we could not get any images that had disconnected vertical black bars with this method.

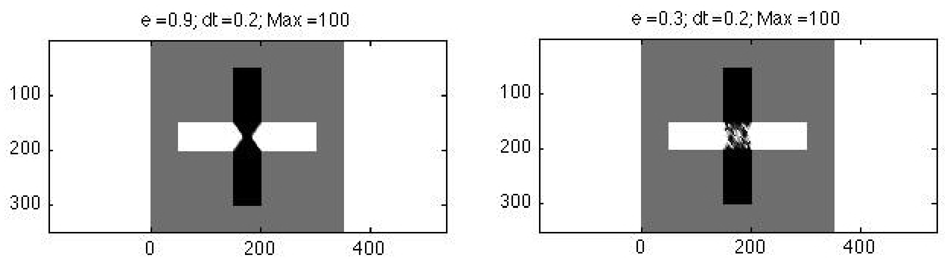

Figure 4.21: Examples of the inpainted image with the fully-implicit method
Left $\bigcirc$: Expected Result, Right $\otimes$: noise

2. Energy plot

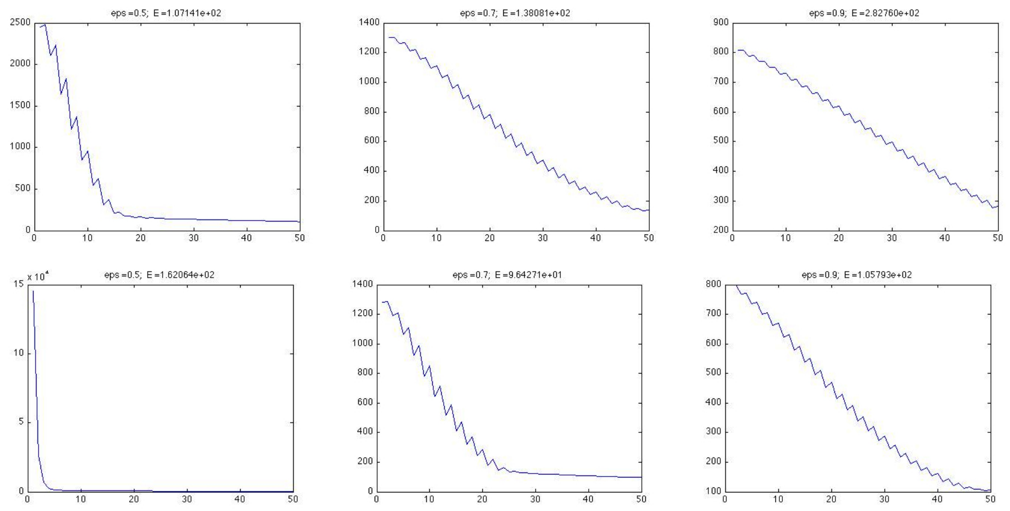

Figure 4.22: Energy Plots for the fully implicit method
Row 1: Iteration = 50, $\Delta t = 0.2$, $\varepsilon = \{0.5, 0.7, 0.9\}$
Row 2: Iteration = 50, $\Delta t = 0.3$, $\varepsilon = \{0.5, 0.7, 0.9\}$,

	values of ε / # of iterations	0.1	0.3	0.5	0.7	0.9	1
$\Delta t = 0.05$	100	-	○	■	■	■	■
$\Delta t = 0.1$	50	-	⊗	■	■	■	■
	100	-	⊗	○	■	■	■
$\Delta t = 0.2$	50	-	⊗	○	○	■	■
	100	-	⊗	○	○	○	○
$\Delta t = 0.3$	50	-	⊗	⊗	○	○	■
	100	-	⊗	⊗	○	○	○
$\Delta t = 0.5$	50	-	⊗*	⊗	⊗	○	○

Notation on the table:
○:= expected result, ■ := grey square at the centre,
⊗:= noise, − : = no change
*:= The image is brighter.

Table 4.4: Results of 4.4.3

Summary:

- We can use larger Δt with the fully implicit method than with the explicit method.
- With the fully-implicit method, we obtain more expected results than with the semi-implicit method.
- The computation time for the fully-implicit method is even longer than for the semi-implicit method.

4.4 Explicit method versus Implicit method

In this section, we will investigate differences in the inpainted images using the explicit and the implicit method.

For the explicit and implicit methods, we use $\Delta t = 0.1$, and $\varepsilon = 0.7$.

1. The explicit method

   We notice that the vertical black bar is disconnected at the centre with 100 iterations, and the final energy is about 180. If we iterate 500 times, the final energy drops to 175. It means by 400 more iterations, the energy changed only 5 units. Therefore, we conclude that after around 100 times iterations, the inpainted image reaches a stable condition. If we iterate 1000 times, the energy goes down further. When we look at the left bottom image in Figure 4.23, the image is not blurred, but the black vertical line is now connected.

2. The semi-implicit method

   We notice that fewer iterations are required for the implicit method. Now, with the semi-implicit method, we notice that the black vertical bar is connected in the middle. (In this experiment, we could not obtain any inpainted images whose vertical black bars were disconnected.) The output images with 60 and 100 iterations look similar. If we iterate further until 200 times, the energy goes down further, but the output image gets blurred.

3. The fully-implicit method

   With the fully-implicit method, the inpainted image has not reached a stable condition after 100 iterations. Hence, in this case, the rate of convergence is slower with the fully-implicit method than with the semi-implicit method. When we iterate 200 times, it seems that it has reached a stable condition. If we iterate further until 300 times, the energy goes down further, but the output image starts getting blurred.

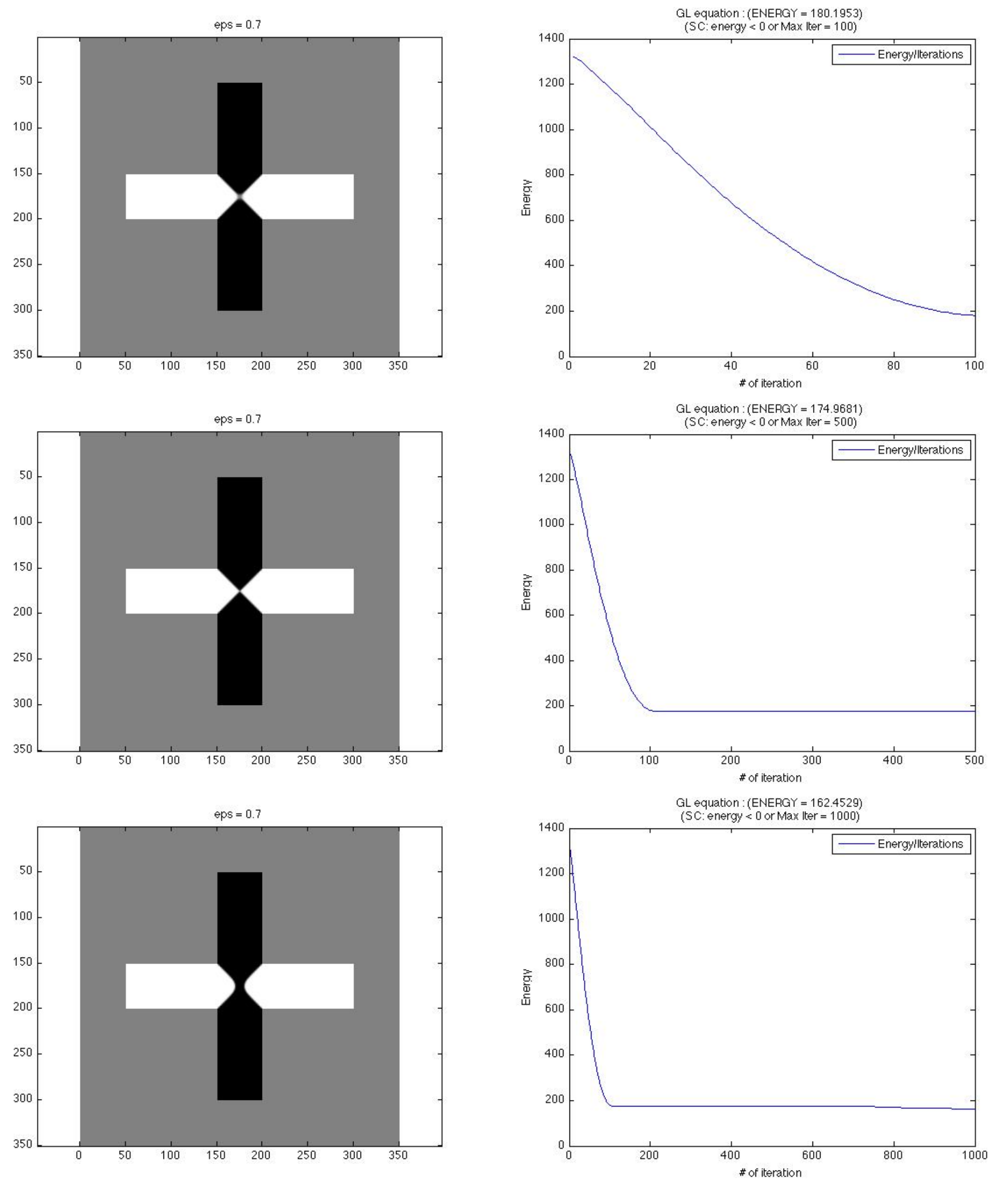

Figure 4.23: **Explicit** method Left : Inpainted image with $\Delta t = 0.1$, $\varepsilon = 0.7$, Top: 100, Middle: 500, Bottom: 1000 iterations Right: The energy of the image

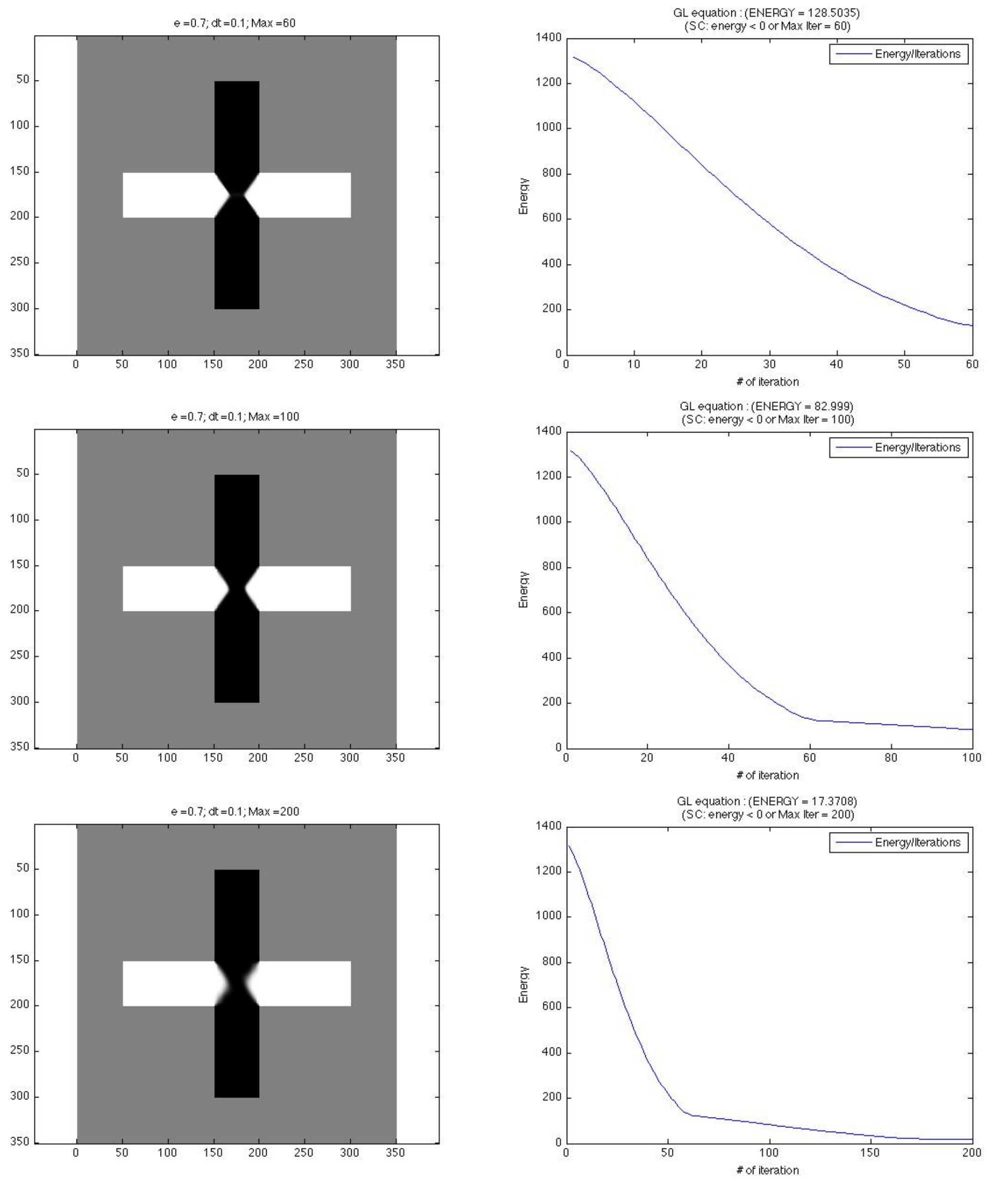

Figure 4.24: **Semi-Implicit** method Left : Inpainted image with $\Delta t = 0.1$, $\varepsilon = 0.7$, Top: 60, Middle: 100, Bottom: 200 iterations Right: The energy of the image

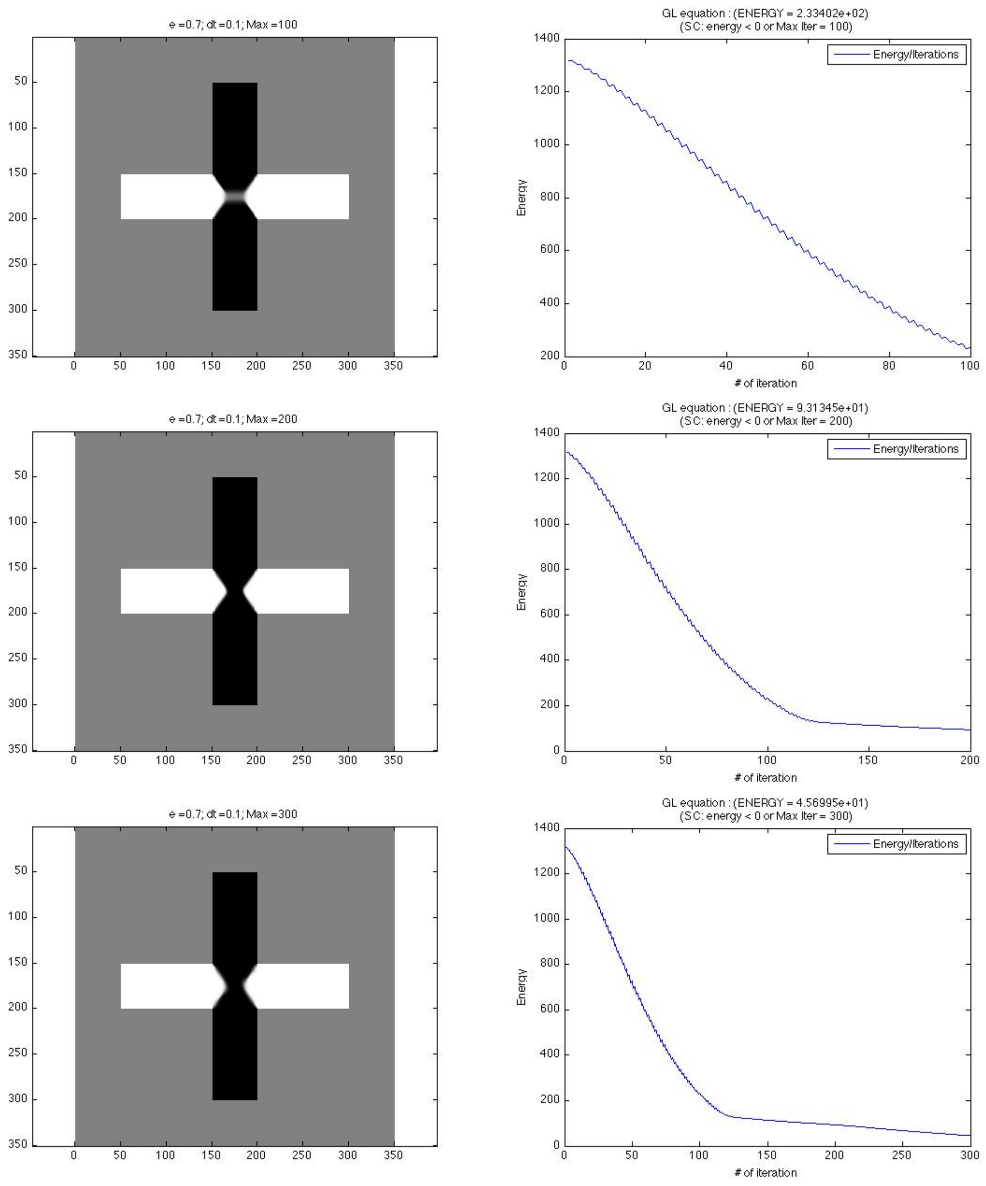

Figure 4.25: **Fully Implicit** method Left : Inpainted image with $\Delta t = 0.1$, $\varepsilon = 0.7$, Top: 100, Middle: 200, Bottom: 300 iterations Right: The energy of the image

Chapter 5

Comparison with the BSCB method

The BSCB method is based on the paper [11]. The BSCB algorithm performs the following steps.

1. The structure of the area surrounding Ω is continued into the gap, contour lines are drawn via the prolongation of those arriving at $\partial\Omega$.

2. The different regions inside Ω, as defined by the contour lines, are filled with colour, matching those of $\partial\Omega$. ([11])

The codes are written based on the discretization of the algorithm given in [11]. In the codes, the anisotropic diffusion (5.1) and the inpainting equation (5.2)

$$\frac{\partial I}{\partial t}(x,y,t) = g_\epsilon(x,y)\kappa(x,y,t)|\nabla I(x,y,t)|, \qquad \forall(x,y) \in \Omega^\epsilon \tag{5.1}$$

$$I^{n+1}(i,j) = I^n(i,j) + \Delta t I_t^n(i,j), \qquad \forall(i,j) \in \Omega \tag{5.2}$$

are implemented alternately.

5.1 Assumptions for Implementation

Since [11] does not go into detail for the implementation, in order to implement the BSCB algorithm, we make assumptions which are explained below. Our results are not necessarily the same as obtained in other implementations, in particular the original paper [11].

5.1.1 The Process of Inpainting

1. For each iteration of the inpainting process, the region under the mask is extracted from $I_t^n(i,j)$, and added to $I_t^{n+1}(i,j)$.
2. As a preprocessing step, the whole original image undergoes the anisotropic diffusion smoothing ([11]). As any specific number is not given in [11], this process is repeated 3 times.

5.1.2 A Smooth Function g for the Anisotrpic Diffusion

The details of the implementation of a smooth function g for the anisotropic diffusion are not given in [11]. We define g as follows for this experiment:

1. The region Ω is dilated with $\epsilon = 4$. In order to make $\epsilon = 4$, the dilation is repeated 3 times with the structure element of disk with radius 1.
2. In order to make a smooth function g, the following values are assigned to each region.

(a)	Ω	: 1.0 (white region)
(b)	Region between Ω and the first dilation	: 0.75
(c)	Region between the first and the second dilation	: 0.50
(d)	Region between the second and the third dilation	: 0.25
(e)	Outside of the dilated region	: 0.0 (black region)

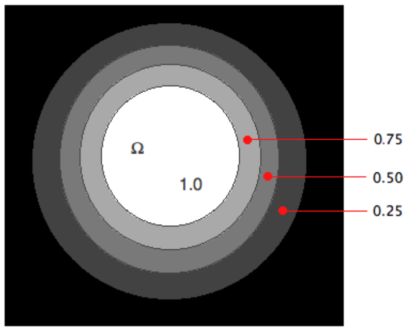

Figure 5.1: Smooth function g

5.1.3 The Euclidean curvature function κ for the Anisotrpic Diffusion

Since the details of the implementation for the Euclidean curvature function κ is not given in [11], we define κ as follows for this experiment.

$$\kappa = \frac{\ddot{I}_{xx}\dot{I}_y^2 - 2\dot{I}_x\dot{I}_y\ddot{I}_{xy} + \ddot{I}_{yy}\dot{I}_x^2}{(\dot{I}_x^2 + \dot{I}_y^2)^{\frac{3}{2}}}$$

where,

$$\begin{aligned}
\dot{I}_x(i,j) &= \frac{(I(i+1,j) - I(i-1,j))}{2h}, \\
\dot{I}_y(i,j) &= \frac{(I(i,j+1) - I(i,j-1))}{2h}, \\
\dot{I}_{xx}(i,j) &= \frac{(I(i+1,j) - 2I(i,j) + I(i-1,j))}{h^2}, \\
\dot{I}_{yy}(i,j) &= \frac{(I(i,j+1) - 2I(i,j) + I(i,j-1))}{h^2}, \\
\dot{I}_{xy}(i,j) &= \frac{(I(i+1,j+1) - I(i-1,j+1) - I(i+1,j-1) + I(i-1,j-1))}{4h},
\end{aligned}$$

with h is the spacial step size ([8]).

5.2 Comparison in Programming

In this section, we look at the programming issues in the Ginzburg-Landau equation and the BSCB algorithm.

5.2.1 The equations to be defined for the BSCB algorithm

First, the discrete inpainting equation

$$I^{n+1}(i,j) = I^n(i,j) + \Delta t I_t^n(i,j), \qquad \forall (i,j) \in \Omega \tag{5.2}$$

consists of

- $I_t^n(i,j) = \Big(\overrightarrow{\delta L^n}(i,j) \cdot \dfrac{\overrightarrow{N}(i,j,n)}{|\overrightarrow{N}(i,j,n)|}\Big)|\nabla I^n(i,j)|$ (5.3)
- $\overrightarrow{\delta L^n}(i,j) = (L^n(i+1,j) - L^n(i-1,j), L^n(i,j+1) - L^n(i,j-1))$ (5.4)
- $L^n(i,j) = I_{xx}^n(i,j) + I_{yy}^n(i,j)$ (5.5)
- $\dfrac{\overrightarrow{N}(i,j,n)}{|\overrightarrow{N}(i,j,n)|} = \dfrac{(-I_y^n(i,j), I_x^n(i,j))}{\sqrt{(I_x^n(i,j))^2 + (I_y^n(i,j))^2}}$ (5.6)
- $\beta^n(i,j) = \overrightarrow{\delta L^n}(i,j) \cdot \dfrac{\overrightarrow{N}(i,j,n)}{|\overrightarrow{N}(i,j,n)|}$ (5.7)

  and

- $|\nabla I^n(i,j)| = \begin{cases} \sqrt{(I_{xbm}^n)^2 + (I_{xfM}^n)^2 + (I_{ybm}^n)^2 + (I_{yfM}^n)^2}, \text{when } \beta^n > 0 \\ \\ \sqrt{(I_{xbM}^n)^2 + (I_{xfm}^n)^2 + (I_{ybM}^n)^2 + (I_{yfm}^n)^2}, \text{when } \beta^n < 0 \end{cases}$ (5.8)

where, the subindices b and f denote backward and forward differences, respectively, while the subindices m and M denote the minimum or maximum, respectively, between the derivative and zero.

The anisotropic diffusion equation

$$\frac{\partial I}{\partial t}(x,y,t) = g_\epsilon(x,y)\kappa(x,y,t)|\nabla I(x,y,t)|, \qquad \forall(x,y) \in \Omega^\epsilon \tag{5.1}$$

where Ω^ϵ is a dilation of Ω with a ball of radius ϵ, consists of

- κ is the Euclidean curvature of the isophotes of I, (5.9)

  and

- g_ϵ is a smooth function in Ω^ϵ such that $\begin{cases} g_\epsilon(x,y) = 0 \quad \in \partial\Omega^\epsilon \\ \\ g_\epsilon(x,y) = 1. \quad \in \Omega \end{cases}$ (5.10)

We have to code all these equations in order to use the BSCB algorithm.

5.2.2 The equations to be defined for the Ginzburg-Landau Equation

In order to run the Ginzburg-Landau method with the explicit discretization scheme, what we have to code is:

$$u_{i,j}^{n+1} = u_{i,j}^n + \Delta t[u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n - 4u_{i,j}^n] + \frac{\Delta t}{\varepsilon^2}\Big[(1 - |u_{i,j}^n|^2)u_{i,j}^n\Big].$$

Therefore, if we only look at the programming issue, the Ginzburg-Landau equation is much easier to implement than the BSCB algorithm.

5.3 Comparison of Inpainted Images

5.3.1 Experiment 1

Now, we compare the inpainted images using the BSCB algorithm and the Ginzburg-Landau equation. Since the BSCB algorithm prolongs the contour lines, the contour lines should be somehow prolonged in the "expected" results.

1. The ambiguous image used in Section 4.3

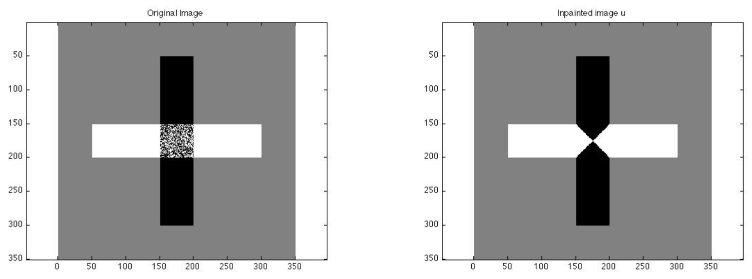

Figure 5.2: Left : Ambiguous image, Right: via Ginzburg-Landau algorithm **u** from Section 4.3

The experiments are done with the BSCB algorithm for $\Delta t = \{0.01, 0.05, 0.1, 0.2\}$ and 1000 iterations. When we look at Table 4.2, we notice that for the Ginzburg-Landau equation, with $\Delta t = 0.01$ and 1000 iterations, we get expected results. And also with $\Delta t = 0.05$ and 500 iterations, except $\varepsilon = 0.01$ and 0.1, we get expected results. But with the BSCB algorithm with 1000 iterations, we do not see any contour lines prolonged. Therefore, we do not get an "expected" inpainted result.

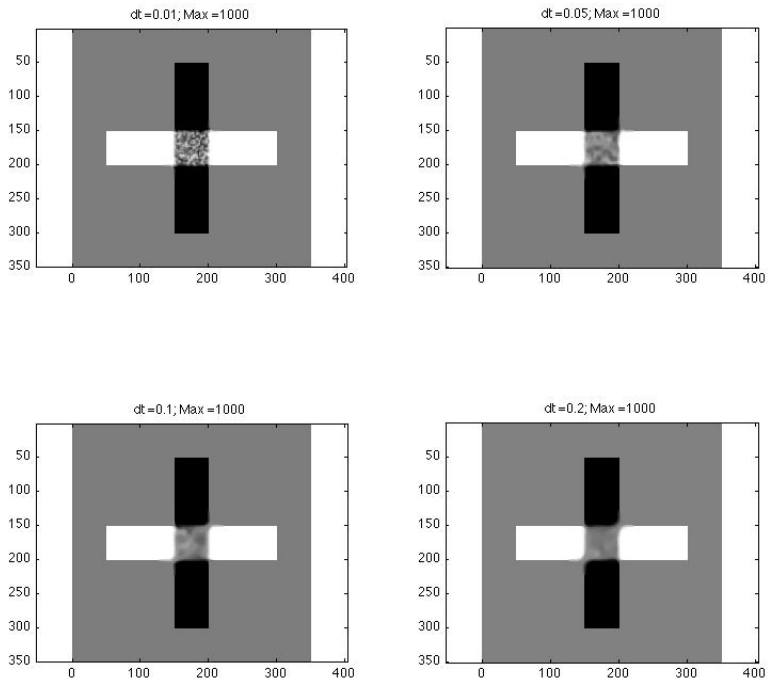

Figure 5.3: The inpainted images of Ambiguous image by BSCB algorithm after 1000 iterations, Top: $\Delta t = 0.01$ (left), 0.05 (right),   Bottom: $\Delta t = 0.1$(left), 0.2 (right)

2. The images with a vertical white bar

   This experiment is done with $\Delta t = 0.1$.

   (a) Line width 50, $\Omega =$ height 150, width 300 in Figure 4.5

   Since (the height of $\Omega = 150$) $<$ (the width of $\Omega = 300$)$-$ (the line width $= 50$), for the Ginzburg-Landau equation with 1000 iterations, we recover the white vertical line. But for the BSCB algorithm, even we if iterate 5000 times, we cannot recover the white vertical line.

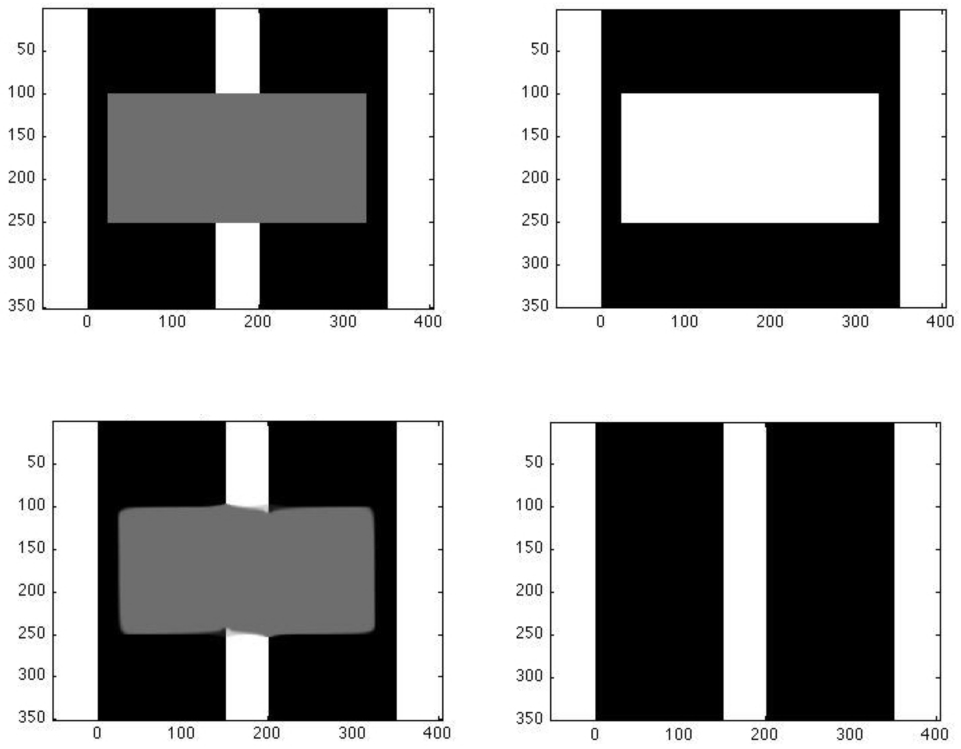

Figure 5.4: LW 50, Ω = height 150, width 300
Top Left: Original Image, Right: Mask Image
Bottom Left: via **BSCB** with 5000 iterations, Right: via **GL equation** with 1000 iterations

(b) Line width 50, $\Omega =$ height 310, width 100 in Figure 4.7
Since (the height of $\Omega = 310$) $\not<$ (the width of $\Omega = 100$)$-$ (the line width $= 50$), for the Ginzburg-Landau equation with 700 iterations, we do not recover the white vertical line. For the BSCB algorithm, even if we iterate 5000 times, also we cannot recover the white vertical line. But we notice that the Ginzburg-Landau scheme has inpainted the vertical line with black colour, whereas, for the BSCB algorithm, we have still Ω to be inpainted.

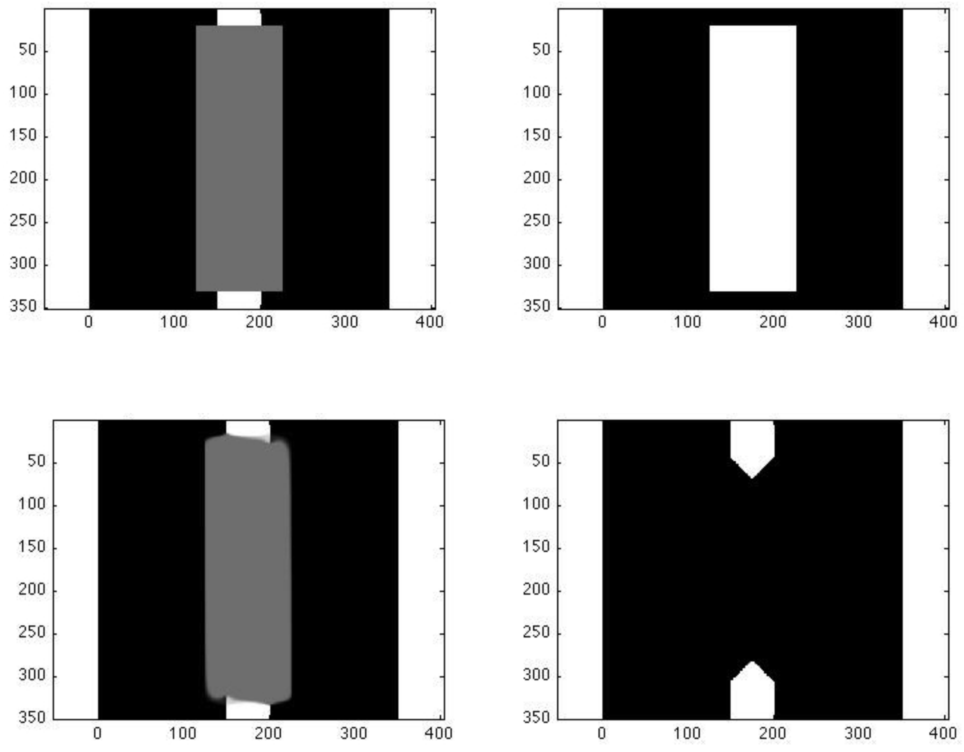

Figure 5.5: LW 50, $\Omega =$ height 310, width 100
Top Left: Original Image, Right: Mask Image
Bottom Left: via **BSCB** with 5000 iterations, Right: via **GL equation** with 700 iterations

3. The images with several levels of brightness
This experiment is done with $\Delta t = 0.1$.

(a) Line width 250, $\Omega =$ height 48, width 300 in Figure 4.10
Since (the height of $\Omega = 48$) $<$ (the width of $\Omega = 300$)$-$ (the line width $= 250$),

for the Ginzburg-Landau equation with 600 iterations, we recover the vertical white and grey lines. But for the BSCB algorithm, even if we iterate 5000 times, we cannot recover the lines.

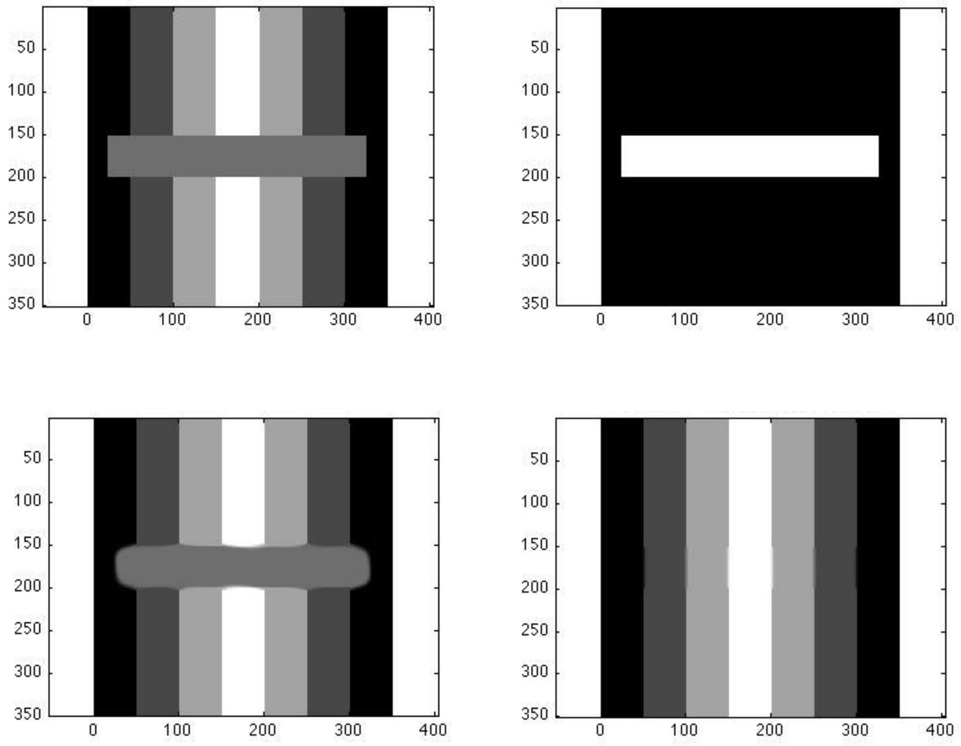

Figure 5.6: Ω = height 48, width 300
Top Left: Original Image, Right: Mask Image
Bottom Left: via **BSCB** with 5000 iterations, Right: via **GL equation** with 600 iterations

(b) Line width 250, Ω = height 310, width 100 in Figure 4.12
Since (the height of Ω = 310) $\not<$ (the width of Ω = 100)$-$ (the line width = 250), for the Ginzburg-Landau equation with 600 iterations, we do not recover the white vertical line in the centre. For the BSCB algorithm, even if we iterate 5000 times, we cannot recover the vertical line. But we notice that the Ginzburg-Landau scheme has inpainted the white line in the centre with grey colour, whereas, for the BSCB algorithm, we still have a large Ω to be inpainted.

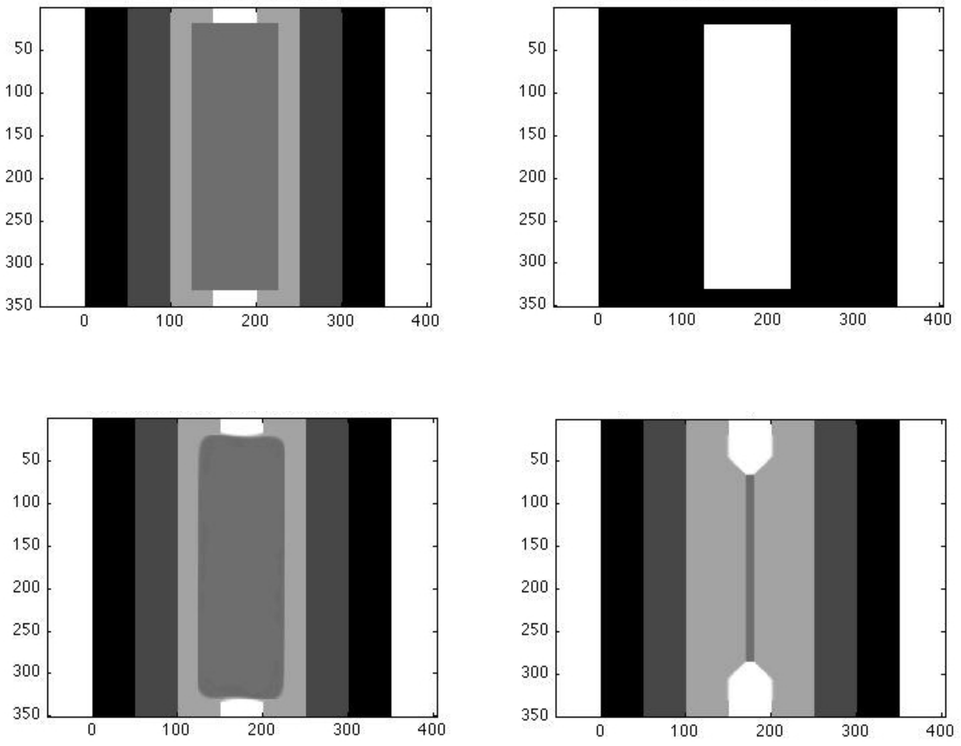

Figure 5.7: Ω = height 310, width 100
Top Left: Original Image, Right: Mask Image
Bottom Left: via **BSCB** with 5000 iterations, Right: via **GL equation** with 600 iterations

5.3.2 Experiment 2

From the results of Experiment 1 , we notice that the rate of convergence is slower for the BSCB algorithm than for the Ginzburg-Landau equation. We will see how the convergence rate differs by using a small Ω in the first example and an actual image in the second example.

- The first example
  We use the image with the vertical line width 20 and Ω of the height 20 and the width 50. For both the Ginzburg-Landau equation and the BSCB algorithm, $\Delta t = 0.1$ is used, and for the Ginzburg-Landau equation, $\varepsilon = 0.5$ is used.

  The result is shown in Figure 5.8. For the Ginzburg-Landau equation, we iterated 100 times, and Ω is inpainted as we expected. For the BSCB algorithm, we iterated 5000 times, and still there is a small Ω to be inpainted. At least the Ginzburg-Landau equation is 50 times faster to inpaint than the BSCB algorithm.

- The second example
  We use the girls image in the second example. For both methods, $\Delta t = 0.1$, and for the Ginzburg-Landau equation $\varepsilon = 0.5$ are used.

  The result is shown in Figure 5.9. For the Ginzburg-Landau equation, we iterated 50 times, and the image is inpainted pretty well. For the BSCB algorithm, we iterated 500 times, and still there is a grey line in the face of the middle girl. The Ginzburg-Landau equation is at least 10 time faster for inpainting than the BSCB algorithm.

We conclude that the Ginzburg-Landau equation with explicit discretization is easier to implement, and faster to inpaint than the BSCB algorithm.

5.3.3 Speeding up the BSCB method via methods in Fluid Dynamics

To increase the speed of the BSCB algorithm, a method based on the Navier-Stokes equations for Fluid Dynamics was introduced by Bertalmio, Bertozzi and Sapiro ([10]). Our comparisons only refer to the original algorithm, and does not take into account the improvements in speed of [10].

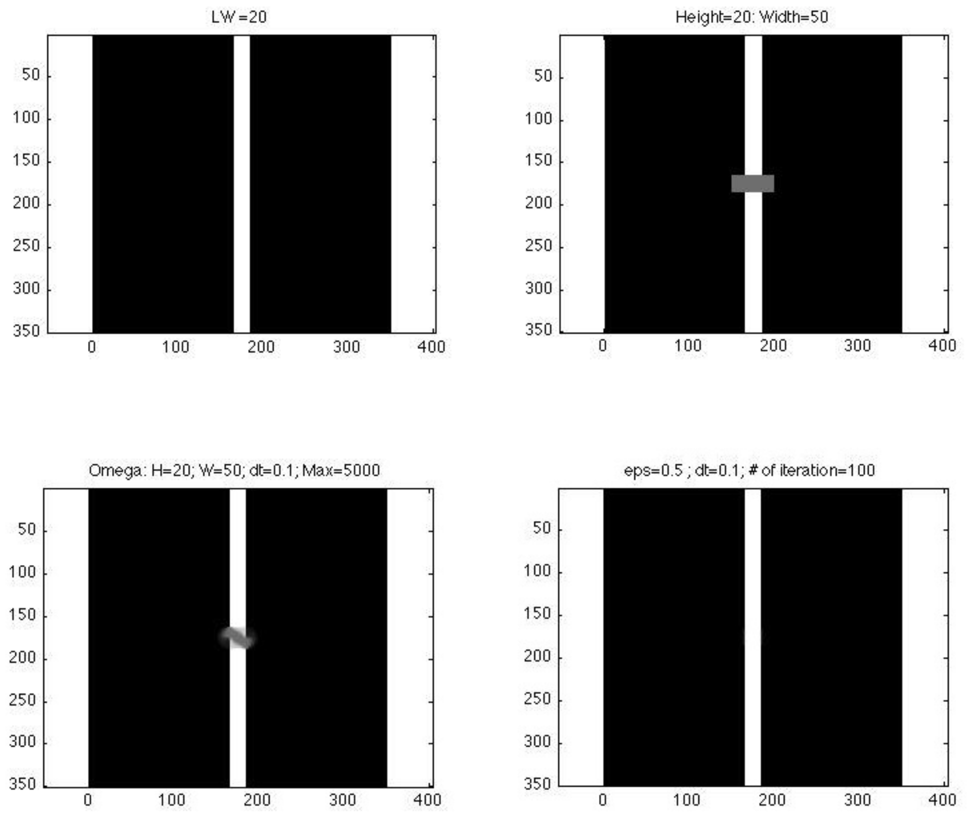

Figure 5.8: The First example
Top: Left Original image with line width 20, Right Ω is grey region
Bottom: Left via **BSCB algorithm**, Right via **Ginzburg-Landau equation**

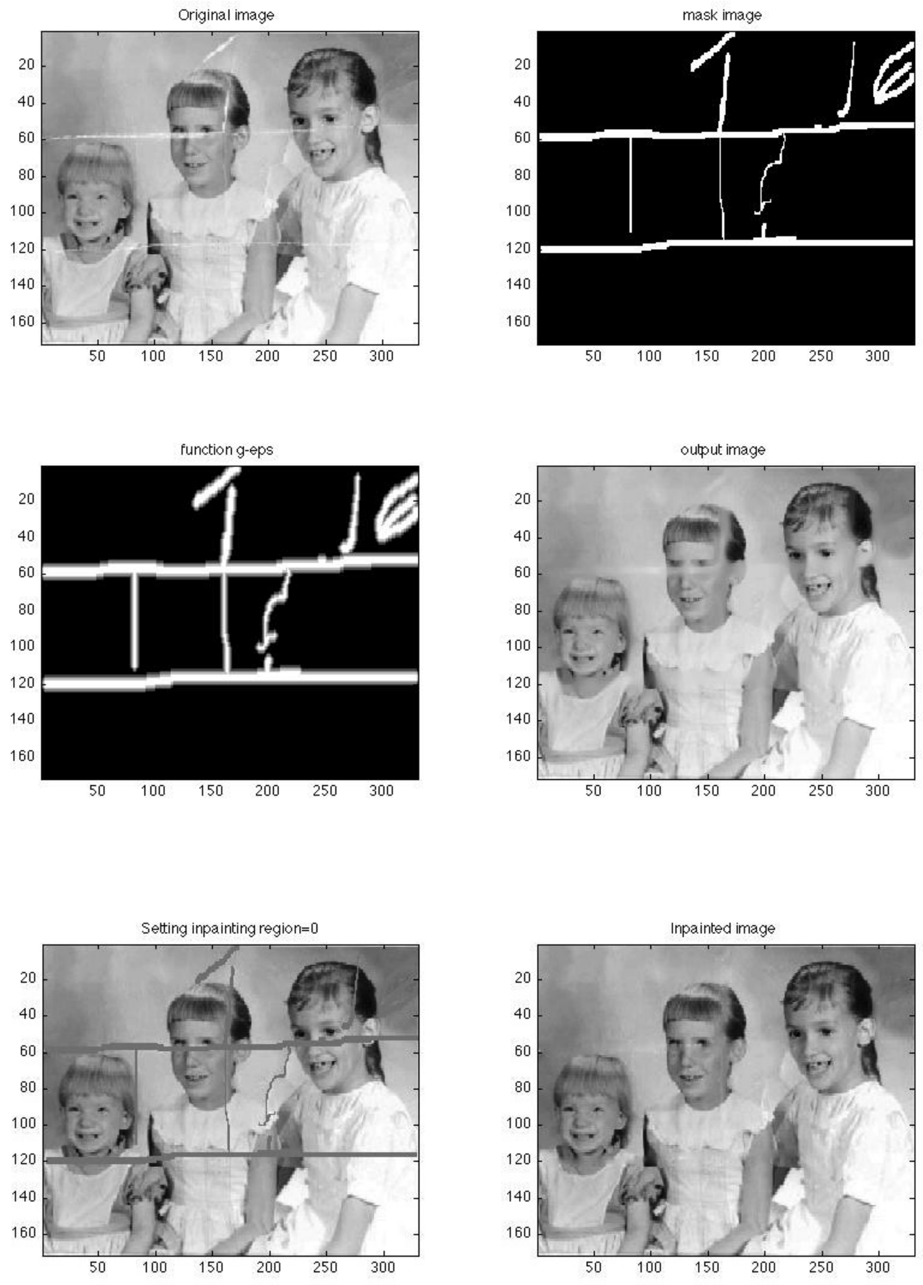

Figure 5.9: The Second example
Top: Left Original image, Right Mask image
Middle: Left Function g, Right Inpainted image via BSCB algorithm with 500 iterations,
Bottom: Left Ω set to be 0, Right Inpainted image via Ginzburg-Landau with 50 iterations

Bibliography

[1] online from $http://www.math.ucla.edu/~imagers/htmls/inp.html$.

[2] online from $http://mathworld.wolfram.com/HyperbolicTangent.html$.

[3] online from $http://www.boopidy.com/aj/cs/inpainting/$.

[4] online from $http://www.cs.berkeley.edu/~aj/cs/inpainting$
No longer available on February 25 2010.

[5] online from $http://www.greyc.ensicaen.fr/~jfadili/demos/WaveRestore$ $/EMInpaint/index.html$.

[6] online from $http://en.wikibooks.org/wiki/Partial\_Differential\_Equations$ $/The\_Laplacian\_and\_Laplace\%27s\_Equation$.

[7] online from $http://en.wikipedia.org/wiki/Phase\_transition\#Order\_parameters$.

[8] online note from $http://www.bioen.utah.edu/wiki/images/6/6f/$ $YounesLectureNotes.pdf$
Mathematical Image Analysis AMS493, Laurent Younes, Johns Hopkins University.

[9] Coloma Ballester, M. Bertalmio, V. Caselles, Guillermo Sapiro, and Joan Verdera. Filling-In by Joint Interpolation of Vector Fields and Gray Levels. *IEEE Transactions on Image Processing*, 10(8):1200–1211, 2001.

[10] M Bertalmio, A.L. Bertozzi, and G Sapiro. Navier-stokes, Fluid Dynamics, and Image and Video Inpainting. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference*, 2001.

[11] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image Inpainting. *SIGGRAPH*, 2000.

[12] Fabrice Bethuel, Haïm Brezis, and Frédéric Hélein. Asymptotics for the minimization of a Ginzburg-Landau functional. *Calculus of Variations and PDE 1*, pages 123–148, 1993.

[13] Fabrice Bethuel, Haïm Brezis, and Frédéric Hélein. *Ginzburg-Landau Vortices.* Birkhäuser, 1994.

[14] Tony F. Chan and Jianhong Shen. Non-Texture Inpainting by Curvature-Driven Diffusions (CDD). *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.

[15] Tony F. Chan and Jianhong Shen. Mathematical Models for Local Nontexture Inpaintings. *SIAM Journal of Applied mathematics*, 62(3):1019–1043, 2002.

[16] Selim Esedoglu and Jianhong Shen. Digital inpainting based on the Mumford-Shah-Euler image model. *European Journal of Applied Mathematics*, 13:353–370, 2002.

[17] Marvin J. Forray. *Variational Calculus in Science and Engineering.* McGraw-Hill, 1968.

[18] V.L. Ginzburg and L.D. Landau. On the Theory of Superconductivity. *Zh. Eksp. Teor. Fiz.(ZhETF) 20*, pages 1064–, 1950.

[19] Harald Grossauer. Digital Inpainting Using the Complex Ginzburg-Landau Equation. Institute of Computer Science Technikerstr. 25 A-6020 Innsbruck AUSTRIA, harald.grossauer@uibk.ac.at.

[20] Harald Grossauer and Otmar Scherzer. Using the Complex Ginzburg-Landau Equation for Digital Inpainting in 2D and 3D. *Scale-Space*, pages 225–236, 2003.

[21] R.M. Hervé and M. Hervé. Étude qualitative des solutions réelles d'une équation différentiell liée à l'équation de Ginzburg-Landau. *Ann. Inst. Henri Poincaré, Anal. Non Linéaire*, 11(4):427–440, 1994.

[22] K.-H. Hoffmann and Q. Tang. *Ginzburg-Landau Phase Transition Theory and Superconductivity.* Birkhäuser, 2001.

[23] Dominic W. Jordan and Peter Smith. *Nonlinear Ordinary Differential Equations An introduction for Scientists and Engineers.* Oxford University Press, 2007.

[24] Randall J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations.* SIAM, 2007.

[25] Simon Masnou. Disocclusion : a variational approach using level lines. *IEEE Transactions on Image Processing*, 11(2):68–76, 2002.

[26] Simon Masnou and Jean-Michel Morel. Level lines based Disocclusion. *International Conference on Image Processing*, 3:259–263, 1998.

[27] Manuel M. Oliveira, Brian Bowen, Richard McKenna, and Yu-Sung Chang. Fast Digital Image Inpainting. *Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain*, pages 261–266, 2001.

[28] Stanley Osher and Ronald Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces.* Springer, 2003.

[29] Frank Pacard and Tristan Rivière. *Linear and Nonlinear Aspect of Vortices The Ginzburg-Landau Model*. Birkhäuser, 2000.

[30] Etienne Sandier and Sylvia Serfaty. *Vortices in the Magnetic Ginzburg-Landau Model*. Birkhäuser, 2007.

[31] J.W. Thomas. *Numerical Partial Differential Equations Finite Difference Methods*. Springer, 1995.