APPLICATIONS OF VARIATIONAL PDE ACCELERATION TO COMPUTER VISION PROBLEMS

A Dissertation
Presented to
The Academic Faculty

By

Minas Benyamin

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Engineering
Department of Electrical and Computer Engineering

Georgia Institute of Technology

May 2022

APPLICATIONS OF VARIATIONAL PDE ACCELERATION TO COMPUTER VISION PROBLEMS

Thesis committee:

Dr. Anthony Yezzi
Department of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Patricio Vela
Department of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Justin Romberg
Department of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Kang Sung Ha
Department of Mathematics
Georgia Institute of Technology

Dr. Aaron Lanterman
Department of Electrical and Computer Engineering
Georgia Institute of Technology

Date approved: April 06, 2022

I have no special talents, I am just passionately curious.

Albert Einstein

In dedication to my Parents, Iman-Sami Zakhari and Michel Nakhla and my loving fiancee, Maryam Botrus

ACKNOWLEDGMENTS

I would like to thank the members of my thesis reading committee Professors Patricio Vela and Justin Romberg for their help in preparation of this work. I would like to especially thank Professor Yezzi, my advisor who aided me immensely throughout the entire PhD process. Without his agreement to be my advisor and subsequent mentoring I likely would have never started a PhD at the Georgia Institute of Technology. I would also like to thank and acknowledge the additional members of my PhD committee Professors Sung Kang Ha and Aaron Lanterman.

I would like to thank and acknowledge Dr. Ganesh Sundaramoorthi. He has been an excellent resource during the time of my PhD program. I would like to thank him for his insights, theoretical discussions and advising. I would also like to thank and acknowledge Dr. Samuel Bignardi for his suggestions and comments. I would like to thank Mr. Geoffrey Goldman and Dr. Tran Luu for their helpful mentoring throughout my years as an undergraduate and their continued career advising in graduate school.

I would like to thank my loving parents who graciously paid many of my bills and gave me their blessing to start a PhD. Without their help I would have been more more financially and emotionally stressed at the prospect of attending a graduate program.

I would like to thank and acknowledge the Smart Scholarship Program for both paying my tuition and supplying my stipend during my years at the Georgia Institute of Technology. Without their financial support doing research at this University would have been far more difficult and financially adverse.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACC Accelerated

ADMM Alternating Direction Method of Multipliers

AWGN additive white Gaussian noise

CFL Courant Friedrichs Lewy

CT Computed Tomography

DFT Discrete Fourier Transform

FFT Fast Fourier Transform

GD Gauss-Seidel

JMIV Journal of Mathematical Imaging and Vision

MS Mumford-Shah

ODE Ordinary Differential Equation

PD Primal-Dual

PDE Partial Differential Equation

ROF Rudin-Osher-Fatemi

SLAM simultaneous localization and mapping

SOR Successive over-relaxation

SP Split-Bregman

TV Total Variation

SUMMARY

This dissertation addresses general optimization in the field of computer vision. In this manuscript we derive a new mathematical framework, Partial Differential Equation (PDE) acceleration, for addressing problems in optimization and image processing. We demonstrate the strength of our framework by applying it to problems in image restoration, object tracking, segmentation, and 3D reconstruction. We address these image processing problems using a class of optimization methods known as variational PDEs. First employed in computer vision in the late 1980s, variational PDE methods are an iterative model-based approach that do not rely on extensive training data or model tuning. We also demonstrate for this class of optimization problems how PDE acceleration offers robust performance against classical optimization methods. Beginning with the most straightforward application, image restoration, we then show how to extend PDE acceleration to object tracking, segmentation and a highly non-convex formulation for 3D reconstruction. We also compare across a wide class of optimization methods for functions, curves, and surfaces and demonstrate that not only is PDE acceleration easy to implement, but that it remains competitive in a variety of both convex and non-convex computer vision applications.

CHAPTER 1
INTRODUCTION

The field of computer vision emerged in the late 1970s with the advent of digital imaging and high performance computing. While initial research focused on the task of image enhancement such as the application of simple filters or correcting pixel errors, [1] the field quickly progressed into a variety of practical applications. Tasks such as object segmentation, tracking and image noise minimization quickly rose to prominence. Later advancements in the field addressed problems such as deblurring, curve driven segmentation, and 3D reconstruction [2]. Many of the early algorithms for image processing are termed as single pass methods. Researchers applied techniques like median or Wiener filters for image correction and kernel methods like match filters for segmentation and object detection. As computer processing power improved, researchers began looking for more powerful approaches for addressing these problems.

In the late 1980s, a major line of work in the field emerged, namely, the application of variational PDE methods for computer vision problems [3]. In variational PDE methods for computer vision, the general approach is to engineer a variational model that allows you to iteratively solve specific image processing problems. These energy-based models take the form of cost functionals which are integrals where the independent variable is itself a function [4]. A classic example of a function often used in computer vision cost functionals would be an image which is not a single variable but has a value at every point (x, y). By careful engineering of the cost functionals, researchers can design them so that the energy, the value of the integral, will be minimized when the desired image processing result is achieved. To minimize these functionals, researchers derive gradient descent PDEs which allow for iterative solutions to these complex image processing problems.

Two of the first applications of variational PDEs in image processing were Perona and Malik's anisotropic diffusion [1], for image denoising, and Mumford-Shah [5], for image segmentation. In their seminal work, Perona and Malik applied the diffusion model to image processing. If we consider diffusion as a physical process, we are modeling the movement of particles from regions of high to low concentration. By allowing diffusion of information in an image we are in effect able to reduce the peaks and troughs in the spatially varying pixel intensity and thus smooth out the noisy pixels. Too much diffusion, however can lead to an over smoothing and a loss of image features. To combat this problem, Perona and Malik proposed a spatially varying diffusion coefficient that encouraged smoothing in regions of the image where pixel values are similar (intraregion) and minimized smoothing in regions where the image has sharp boundaries (interregion).

In the segmentation domain, Mumford and Shah proposed the Mumford-Shah functional which measured the degree of match between an image and its segmentation. The functional is composed of three components: 1) A squared penalty that measures the difference between the segmented image and the original image 2) An L_2 penalty on the variation of the segmented regions 3) An arclength penalty that ensures that the curves that segment the image are as short as possible. One of the strengths of their segmentation model is that the segmenting curves do not have to be connected, only piecewise smooth. This in effect allows for the segmentation of disconnected regions in the image and allows the method to simultaneously segment multiple regions while still minimizing the energy. Another strength of this method is that by increasing the penalty on the arclength, it can reduce the capture of noise or spurious details in the image.

The class of variational PDE methods continued to increase in popularity [6] and were among the state of the art approaches until the feasibility of training data driven models such as those used in machine learning and neural networks. However, in absence of labeled training sets and robust approaches for transfer learning, they remain among the most successful methods in image processing.

One of the drawbacks of the initial class of variational methods, however, was the large number of iterations required for gradient descent. As a result of this performance limitation, researchers increasingly looked to formulate convex cost functionals with a single global minimum. This guarantees that gradient descent will converge towards the minimum energy and permits the use of an entire class of fast optimization methods. Techniques such as Primal Dual, Split-Bregman (SP) and Alternating Direction Method of Multipliers (ADMM) can quickly and efficiently solve convex optimization problems without the need for model training or excessive iterations [7, 8, 9]. And while these methods are incredibly successful they are not without drawbacks as they often require a very careful formulation of the cost functional as to ensure that the subsequent minimization is convex. More complex problems such as variational 3D reconstruction, and radar and acoustic-based shape reconstruction are often non-convex and can have local minima in the energy function. In addition, because there is not necessarily a single minimum energy, global minimization methods such as ADMM, Primal-Dual (PD), and SP are not as readily applicable. And although gradient descent can be applied to these problems the descent is often slow and prone to getting trapped in local minimizers.

To address these problems, we have developed an accelerated geometric gradient descent method, PDE acceleration. This framework, first proposed by Anthony Yezzi and Ganesh Sundaramoorthi [10], works by incorporating the momentum of the previous step into the current descent update. The concept is similar to a ball rolling down a hill; see Figure 1.1a. In typical gradient descent, the ball is stopped, the gradient of the hill is checked and the ball moves in the direction of the gradient. This stop and start behavior however can trap the ball inside a local minimum. While impos-

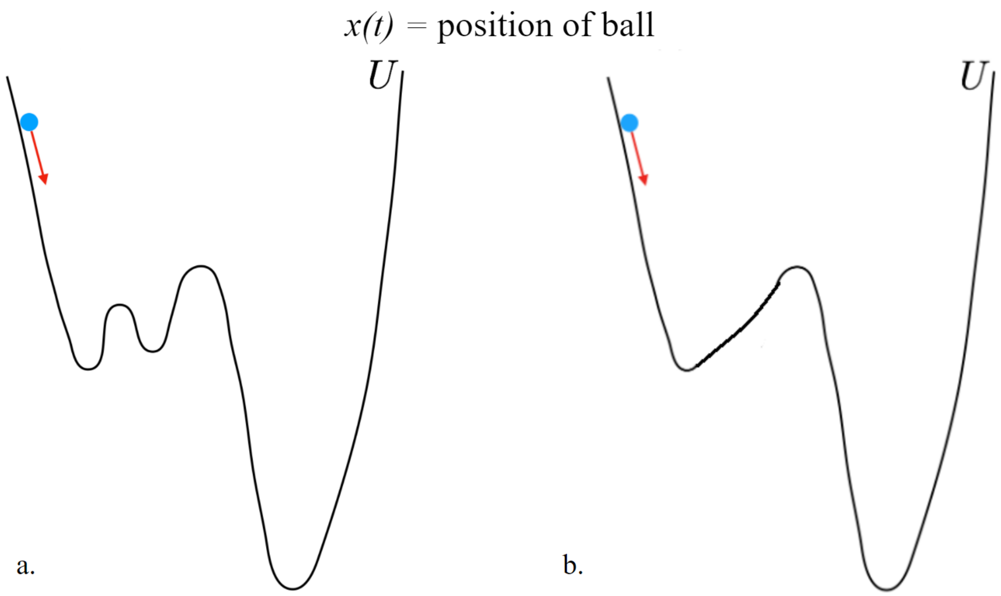


Figure 1.1: Accelerated Descent No Regularization [a] vs Regularization [b]

ing regulation on the energy function can improve the descent, it does not necessarily guarantee the removal of sufficiently large local minimizers; see Figure 1.1b.

To show the application and usefulness of this framework, we have applied PDE acceleration to a broad class of computer vision problems that have been formulated within a variational PDE framework. These image processing problems include inversion problems such as denoising and deconvolution, active contours, optical flow, 3D stereo-reconstruction as well as computational problems such as Poisson solvers.

The early sections of this manuscript describe the background for these image processing problems and also establish the mathematical basis for the framework of PDE acceleration. In this, we detail how the framework is extended from the case of Ordinary Differential Equation (ODE)s to the much more general PDE domain. The applications in which we applied the framework are presented in an order of increasing sophistication and complexity. We begin with an extensive discussion of discretization schemes, stability analysis, and regularizers. From this foundation, we first address the most simple of flat space cases the Rudin-Osher-Fatemi (ROF) [11] denoising functional. Next, PDE acceleration is extended to several suitable candidates for inversion problems including: deconvolution, compressed sensing and as a general Poisson solver. With the most straightforward application of the framework addressed, we move to the case of image registration and diffeomorphisms. For the optical-flow problem, we demonstrate the versatility of PDE acceleration for the Middlebury data set [12] and compare the results against the gold standard conjugate gradient. Then we extend PDE acceleration to the non-convex application of active contour segmentation and active surfaces, often used in 3D reconstruction [13]. Through these applications we demonstrate the most powerful use of PDE acceleration in successfully attacking non-convex optimization problems.

1.1 Organization

This dissertation is split into five separate sections: the introduction, the literature review, applications of PDE acceleration, the conclusion, and the appendices. The first chapter of this manuscript introduces the research area at hand and directs the reader to the focus of this dissertation. We also detail, in the introduction, the principal contributions of this research. Next, the literature review discusses an overview of the relevant image processing problems, within a variational context, and gives examples of how acceleration is used in current optimization. In the applications of PDE acceleration we derive the PDE acceleration framework and present several applications of the framework to different classes of image processing problems. This research includes experiments from four of our accepted publications [14, 15, 16, 17].

1.2 Contributions

The contributions of this research are specifically: **1)** The presentation of a novel framework for accelerating variational PDEs. **2)** The application of this framework for the flat-space case of regularized inversion problems. **3)** The application of this framework to the case of diffeomorphisms and optical flow. **4)** The extension of this framework for the active contour case. **5)** A non-convex application of the framework to the problem of active surfaces, used in 3D reconstruction.

CHAPTER 2
LITERATURE SURVEY

2.1 Variational PDEs

Variational partial differential equations have played an important role in image analysis and computer vision for several decades. First employed in image processing in the 1980s they are computationally more expensive than single-pass methods but are highly adaptable and have the potential to offer far better results. They represent the continuum limit of a class of model-based iterative methods that seek to minimize cost functions through a descent process.

The first step in these problems is to design an energy functional that captures the properties desired in the solution of the image processing problem. These functionals take as input an entire function and typically have the form of an integral over the domain of the input function. The idea is to find an input function that minimizes the functional so that it will provide a good solution for the image processing problem.

In order to minimize a cost functional we calculate its first variation [18] to obtain the descent direction, or the optimal perturbation the input function that maximally decreases the functional. We refer to this as the (negative) gradient of our energy with respect to the evolving function. Once the optimal perturbation is known we can deform the function in this direction using a descent evolution in the form of a system of PDEs [6]. We refer to this as continuum gradient descent.

The next step is to take our continuum gradient descent equations and implement them on a finite dimensional grid. To implement these PDEs in software we use the Euler discretization schemes to write discrete update equations that allow us to update our function in finite time increments of Δt. After deriving our update equations we must also consider the stability of the discrete PDEs. To determine an upper bound for our time step, Δt, we derive the Courant Friedrichs Lewy (CFL) stability conditions.

One drawback of this minimization technique however is that PDE methods for non-convex problems are prone to getting trapped in local minimizers. This problem is especially apparent in highly non-convex formulations such as some active-surface models where even large degrees of regularization can fail to remove the local minima. To address this problem we propose the PDE acceleration framework, which we will use to both accelerate the minimization of cost functionals and to improve on the robustness of their minimizations, specifically that we can now bypass many of the local minimizers in the energy space. We will then examine the benefits of PDE acceleration for cost functionals used in image denoising, deconvolution, optical flow calculation, active contour driven segmentation, and stereo reconstruction.

2.2 Acceleration in Optimization

Numerous versions of accelerated descent and momentum-based schemes have been proposed in the literature for *finite dimensional* problems and are widely used both within the optimization [19, 20, 21, 22] and machine learning communities [23, 24]. A key property for accelerated descent schemes is an incorporation of the previous search direction together with the current gradient into the calculation of the next search direction. As such this inserts memory into the dynamical process.

One of the earliest such techniques is Polyak's heavy ball method [25] which averages the local descent with the previous direction. His work was extended and studied in the continuum case by [26, 27] and is known as the *heavy ball with friction.* One of the most famous advances in momentum that followed Polyak's work is Nesterov's accelerated gradient descent [28]. In this seminal work Nesterov proves an optimal convergence of $O(1/k^2)$ for smooth convex problems with k iterations. This in turn has led to numerous applications of accelerated descent within the study of convex optimization [7, 29]. In particular, Su, Boyd and Candes [30] recently showed that Nesterov acceleration is simply a discretization of the second order ODE

$$\ddot{x} + \frac{3}{t}\dot{x} = -\nabla f(x). \tag{2.1}$$

Other works have since termed this ODE as *continuous time Nesterov* [31]. We note the friction coefficient $3/t$ vanishes as $t \to \infty$, which explains why many implementations of Nesterov acceleration involve *restarting*, or resetting the time to $t = 0$ when the system is underdamped [31].

In what was perhaps the most expansive extension of Nesterov's work, Wibinsono, Wilson and Jordan [32] showed that his accelerated descent methods are discretizations of the ODE equations of motion for a particular Lagrangian action functional. This in effect is a variational generalization of Nesterov's acceleration and allows for its extension into the PDE setting [14, 16, 17, 33, 34, 15]. Wibinsono, Wilson and Jordan's work is summarized as follows. They start with the Bregman divergence of a convex distance generating function h:

$$D_h(y, x) = h(y) - h(x) - \langle \nabla h(x), y - x \rangle \tag{2.2}$$

and utilize careful discretizations of the Euler-Lagrange equation for the time integral (evolution time) of the following Bregman Lagrangian

$$\mathcal{L}(X, V, t) = e^{a(t)+\gamma(t)} \left[D_h(X + e^{-a(t)}V, X) - e^{b(t)}\mathbf{U}(X) \right]$$

where the potential energy $\mathbf{U}$ represents the cost to be minimized[1]. In the Euclidean

[1] The functions a, b, γ are arbitrary smooth continuously differentiable functions that determine the weighting of the velocity, potential and damping of the Lagrangian.

case, where $D_h(y,x) = \frac{1}{2}\|y-x\|^2$, this simplifies to

$$\mathcal{L} = e^{\gamma(t)} \left[e^{-a(t)} \underbrace{\frac{1}{2}\|V\|^2}_{\mathbf{T}} - e^{a(t)+b(t)}\mathbf{U}(X) \right]$$

where $\mathbf{T}$ models the kinetic energy of a unit mass particle in $\mathbb{R}^n$. Nesterov's methods [28, 22, 35, 36, 37, 38] belong to a subfamily of Bregman Lagrangians with the following choice of parameters (indexed by $k > 0$)

$$a = \log k - \log t, \qquad b = k\,\log t + \log \lambda, \qquad \gamma = k\,\log t$$

which, in the Euclidean case, yields a time-explicit *generalized action* (compared to the time-implicit standard action $\mathbf{T} - \mathbf{U}$ from classical mechanics [39]) as follows.

$$\mathcal{L} = \frac{t^{k+1}}{k}\left(\mathbf{T} - \lambda k^2 t^{k-2}\mathbf{U}\right) \tag{2.3}$$

In the case of $k = 2$, for example, the Euler-Lagrange equations for the integral of this time-explicit action yield the continuum limit of Nesterov's accelerated mirror descent [38] derived in both [40] and [20].

2.3 Variational Methods in Image Processing

Variational PDE methods have been applied in image analysis and computer vision for several decades [2, 41, 3]. They are an excellent tool for formulating model-based approaches and have wide utility. In the following sections we provide background for variational approaches to image denoising, deconvolution, optical flow, active contour driven segmentation, and stereoscopic segmentation for 3D reconstruction.

2.3.1 Variational Image Denoising and Deconvolution

In image denoising the objective is to correct the erroneous or corrupted pixels in an image and improve the quality of the picture. In a typical digital camera the picture is created from the light measured at the photodiodes. The signal strength or signal to noise ratio of the image can be described as a function of the number of photons received. As the level of light per diode decreases, such as in low light imaging, the pixel values corresponding to those diodes are increasingly subject to random processes in the imaging sequence. Other sources of image noise can also be introduced at the level of the detector, for example broken photodiodes, or in the analog to digital conversation often in the form of electronic noise. We can approximate this problem through the following signal model:

$$g = h + n \tag{2.4}$$

where g is the received signal, h is the true image we wish to recover, and n is some noise function of unknown distribution. One of the most common and older approaches for addressing this problem is the median filter which uses neighboring pixels to fill in corrupted pixels. The median filter operates with the assumption that the corrupted pixels do not significantly change the median over short distances say 3-5 pixel windows. In practice this works quite well for images with minimal degradation and sufficiently high resolution. To improve upon median filters, one of the first applications of variational PDEs to computer vision was the advent of total variational denoising [11]. The premise behind Total Variation (TV) was that clean images had less total variation than a noisy version of the same image. The ROF model formulated by Rudin, Osher and Fatemi included an L_1 TV norm which they used to minimize spurious signals in the image. The cost functional, when image u

is differentiable, is given here:

$$E(u) = \frac{\lambda}{2}\int_{\Omega}(g-u)^2dx + \int_{\Omega}|\nabla u|dx \tag{2.5}$$

where λ is a tuning parameter used to adjust the amount of regularization and $|\nabla u|$ is the norm of the gradient of the evolving image. This initial model has seen wide adaptation and while initially proposed for additive white Gaussian noise (AWGN) the ROF model has been widely extended to other noise sources as well [42, 43, 44, 45]. Typically, the most obvious approach to minimizing this cost function is using the gradient descent update derived from the first variation [11]. One benefit of this functional is that it lends itself well to convex minimizations [46]. Several prominent approaches for convex optimizations such as Primal Dual [7], Split Bregman [8, 46] and ADMM [47, 9] have arisen in recent years and are generally considered to be the state of the art for denoising problems outside of deep learning based approaches such as [48]. One idea for potentially improving TV denoising is the modification of the regularizing term. For example in older schemes, researchers would often use an L_2 penalty or Tikhonov regularization [49]:

$$E(u) = \int_{\Omega}\frac{\lambda}{2}(u-g)^2 + ||\nabla u||^2 \ \ d\mathbf{x} \tag{2.6}$$

This approach preceded the work of Rudin, Osher and Fatemi, and often yielded visually worse reconstructions. However, it does avoid some of the unintended behavior of the ROF model such as staircasing. Another idea is to create an interplay between an L_1 and L_2 norm by introducing a Beltrami norm:

$$E(u) = \int_{\Omega}\frac{\lambda}{2}(u-g)^2 + \frac{1}{\beta}\sqrt{1+||\beta\nabla u||^2} \ \ d\mathbf{x} \tag{2.7}$$

Here the intent is that for a sufficiently large β the norm would behave as an L_1

norm but that it could be adjusted so as to avoid some of the associated staircasing[2] [50].

2.3.2 Deconvolution

In the deconvolution problem, we are typically trying to reverse the effect of a distortion on our received signal. Physically this problem often occurs because signals must propagate through space from the source to the receiver. Physical phenomena such as the effect of the atmosphere, water vapor and spherical spreading alter the signal as it propagates from the source to the receiver. In image processing severe signal degradation can also result from issues in the imaging system. For example, cameras that have slow shutter speeds are often prone to motion blur because objects in the scene can undergo significant movement during the time when the shutter is open. Mathematically we can describe the effects of the physical world on our received signals using a linear convolution. Consider the standard model [3] in a degraded image:

$$g = K * h + n \tag{2.8}$$

where g is again the received signal, K is the convolution operator that encompasses the channel effect and n is some unknown noise distribution. This inversion problem, trying to undo the effect of the channel, is immediately much more difficult than the case of denoising. Even if a very good estimate for the convolution operator K is obtained attempting to simply deconvolve the image will leave the result with large errors centered at each of the corrupted pixels[3]. One way to address this problem is a two step approach where the image is first denoised and then deconvolved. An alternative approach is to attempt the deconvolution with a strong regularizer

[2] We use both Equation 3.82 and Equation 2.5 for our later experiments.

[3] Additionally the problem becomes even more difficult when K is not known or easily estimated.

that can help minimize the effect of the noise on the final result. For our work we will be considering the case where K is well known or the non-blind problem[4]. Let us first consider the same TV approach as was used for denoising. In this deconvolution problem, we are generally seeking to minimize some norm of the form $||(K * u - g)||$. In the case of TV, with a smooth image u, this gives rise to the following functional [52]:

$$\int_{\Omega}(g - K * u)^2 dx + \beta \int_{\Omega} |\nabla u| dx \tag{2.9}$$

where the regularizer[5] with coefficient β constrains the smoothness of the deconvolved image. As in the case of Equation 2.5, several popular convex minimizations of Equation 2.9 have risen to prominence in the literature and include ADMM, Primal Dual, and Split Bregman [53, 7, 54] among others. Another recent approach [55, 51, 3] has been to replace the TV norm with the edge preserving regularizers used in the Mumford-Shah (MS) functional[6]. These penalty terms provide an improved restoration of edges over TV but often leads to the presence of artificial ridges and peaks in the restored images. One alternative to both the TV and MS regularizers is to use a Beltrami norm [50]. For Equation 2.9 and Equation 2.5 this would entail a simple replacement of the regularizer by $\frac{1}{\beta}\sqrt{1 + ||\beta \nabla u||^2}$. One of the benefits of this regularizer is that the gradient descent update is generally better behaved and is known to work well [50]. In the formulation of the second order update equation Equation 3.82 and in [14] we choose this as a candidate for PDE acceleration because of the success this norm had already seen with first order schemes.

[4] While we do not consider it in depth here, the case of blind or rather semi-blind restoration is of significant interest in the community; these problems are often approached by constraining the characteristics of the linear convolution operator [51]. This is typically done by modeling the physics and optics of the imaging device [3].

[5] An earlier approach was to use an L^2 regularizer ; this however led to significant over smoothing of the image [49, 3].

[6] In their seminal 1989 paper [5, 56] Mumford and Shah proposed an energy functional that allowed for the computation of piece-wise smooth iterations u from a given initial state g. The initial applications were for image segmentation.

2.3.3 Variational Optical Flow

A popular problem in image processing is to track an object from one frame to the next. One approach to this is to examine how the brightness pattern in the scene changes from image to image. By resolving how the brightness in a scene is changing we can use this to characterize the movement of the objects we wish to track. This problem, commonly referred to as optical flow, lends itself well to variational formulations [57, 58, 59, 60, 61, 62, 63, 64] where the objective is to calculate the flow or displacement of brightness patterns from one image to the next. Many researchers in the field of computer vision will often refer to both of these calculations as *optical flow*. In practice the shifts and changes in the brightness correspond to a physical displacement of objects in the scene or a movement in the camera [65]. One of the earliest and most well-known variational approaches was proposed by Horn and Schunk [57]. In their seminal paper the authors derive a model that relates the change in image brightness at a point to the motion of the brightness pattern. This gives rise to the following constraint $I_x u + I_y v + I_t = 0$ where I_x, I_y, and I_t are the spatial and time derivatives of the image and u and v constitute the horizontal and vertical components of the optical flow. Horn and Schunk also propose a smoothness constraint where they minimize the sum of the L_2 norms of the flow vector[7]. Their cost functional is provided in Equation 2.10.

$$E = \int\int \Big((I_x u + I_y v + I_t)^2 + \alpha^2 (||\nabla u||^2 + ||\nabla v||^2) \Big) dxdy \tag{2.10}$$

Here α controls the degree of smoothness in the optical flow.

Another popular approach is to penalize the brightness pattern displacement between the first and second image. This second approach was first proposed by Lukas and Kanade and requires computing the forward displacement $\phi(x)$. The general idea

[7] Recall that u and v occupy the same dimensionality as the image.

in the Lukas and Kanade approach is given as follows penalize $\sum_{i,j}[I_1(i+u_{i,j}, j+v_{i,j}) - I_0(i,j)]$ where i, j are the indices of each pixel and I_1 and I_0 are the images in which we desire to compute the brightness pattern displacement [66, 67]. One advantage of this approach is that it allows for the calculation of larger brightness shifts than Horn and Schunk. For our experiments in optical flow we used the standard sum of squared differences for this penalty as given in [68] and the second order smoothness constraint imposed by [69, 70]. Writing the forward displacement as $\phi(x) = [u, v]$ allows us to write the form of Equation 2.11[8].

$$U(\phi) = \frac{1}{2}\int_{\mathbb{R}^n} |I_1(\phi(x)) - I_0(x)|^2\, dx + \frac{1}{2}\alpha \int_{\mathbb{R}^n} |\nabla(\phi(x) - x)|^2\, dx \tag{2.11}$$

One of the limitations of most variational optical flow methods is that calculating larger shifts in the brightness are often expensive and slow to converge. In order to have a more fair comparison between PDE acceleration and other methods, we implemented several best-practices to speed up the convergence of the discrete implementations. These best-practices were not necessary for PDE acceleration but provided better comparisons of real-world performance.

One of the most popular best-practices[9] that has emerged is to use an iterative pyramid scheme [67]. In this scheme the idea is to compute the optical flow at lower sub-sampled resolutions of the image (see Figure 2.1) and to then interpolate the flow to the next level of the pyramid. The interpolated flow can then be used to initialize the starting value in the descent scheme at that higher resolution. This saves extensive time in the highest resolutions where the recovery problem is the most challenging due to having the largest brightness pattern displacements. This basic strategy of linearization[10], iterative warping and image pyramids have been

[8] Far more expansive discussions of this narrative and the development of optical flow methods can be found in [67, 71, 64, 65] which we will use in our later experiments.

[9] A more comprehensive list of competing approaches and modern best practices can be found in [67].

[10] In [60, 61] the authors linearize the image intensities. In [59] linearization is avoided but the

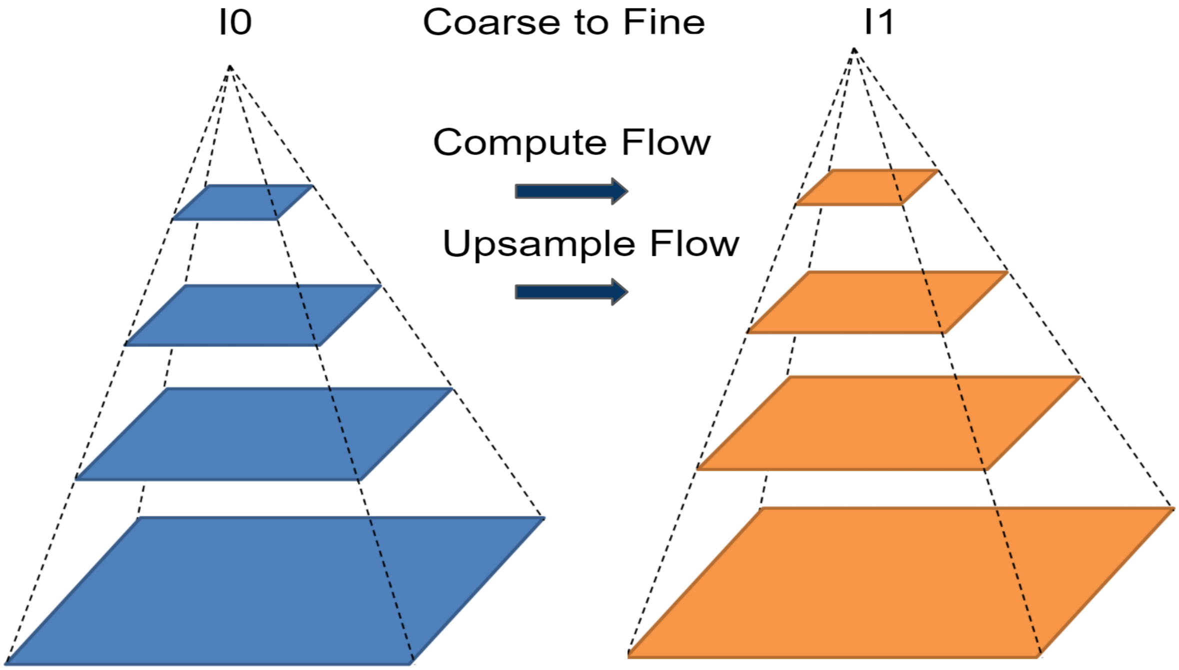


Figure 2.1: Optical Flow pyramid used for up-sampling flow

the dominant approach to many variational optical flow models (e.g., [57, 58, 59, 60, 61]), regardless of the regularization that is used (e.g., use of robust norms, total variation, non-local norms, etc). And despite some progress, most modern optical flow algorithms involve simplification of the minimization into a linear problem and employ computational-optimizations [67] such as up-sampling and intermediate filters.

In our research, however, we constructed accelerated gradient descent algorithms that are applicable to any variational optical flow algorithm and we avoid the linearization step and aim to obtain a better optimizer. Here, we consider the case of optical flow modeled as a global diffeomorphism, but in principle this can be generalized to piecewise diffeomorphisms as in [63]. To evaluate optical flow algorithms it is also necessary to compare[11] on one of the several available data sets used for comparing recovered flow maps [73, 12, 74]. For the experiments presented in section 3.4

authors still employ a similar coarse to fine strategy.

[11] One of the most common methods for computing optical flow and the one we compared against was conjugate gradient [72].

we choose to do our evaluation on the Middlebury[12] data set [12] which is widely used throughout the published literature. We show empirically that our accelerated method can out-perform the standard linearized approach, conjugate gradient, for optical flow in terms of computational speed. We likewise employ the pyramid scheme to better illustrate the performance advantages of PDE acceleration even when utilizing modern optimizations and best practices.

2.3.4 Diffeomorphic Image Registration

This work relates to the literature on diffeomorphic image registration [76, 77, 78, 79], where the goal, similar to ours, is to compute a registration between two images as a diffeomorphism. There a diffeomorphism is generated by a path of smooth velocity fields integrated over time. Rather than formulating an optimization problem directly on the diffeomorphism, the optimization problem is formed on a path of velocity fields. The optimization problem is to minimize $\int_0^1 \|v\|^2 \, \mathrm{d}t$ where v is a time varying vector field, $\|\cdot\|$ is a norm on velocity fields, and the optimization is subject to the constraint that the mapping ϕ maps one image to the other, i.e., $I_1 = I_0 \circ \phi^{-1}$. The minimization can be considered as the minimization of an action integral where the action contains only a kinetic energy. The norm is chosen to be a Sobolev norm to ensure that the generated diffeomorphism (by integrating the velocity fields over time) is smooth. The optimization problem is solved in [76] by a Sobolev gradient descent on the *space of paths*. The resulting path is a geodesic with Riemannian metric given by the Sobolev metric $\|v\|$. In [77], it is shown these geodesics can be computed by integrating a forward evolution equation, determined from the conservation of momentum, with an initial velocity.

Our framework instead uses accelerated gradient descent. Like the work of [76, 77], it is derived from an action integral, but the action has both a kinetic energy

[12] In our comparison the two recovery benchmarks are average angular error and end point error. These reported errors are two of the most common to encounter in the field of optical flow [75].

and a *potential* energy, which is the objective functional that is to be optimized. In this current work, our kinetic energy arises naturally from physics rather than a Sobolev norm. One of our motivations in this work is to get regularizing effects of Sobolev norms without using Sobolev norms, since that requires inverting differential operators in the optimization, which is computationally expensive. Our kinetic energy is an L^2 metric weighted by *mass*. Our method has acceleration, rather than zero acceleration in [76, 77], and this is obtained by endowing a diffeomorphism with mass, which is a mass density in $\mathbb{R}^n$. This mass allows for the kinetic energy to endow the optimization with dynamics. Our optimization is obtained as the stationary conditions of the action with respect to both velocity and *mass density*. The latter links our approach to optimal mass transport, described earlier.

2.3.5 Connections to Sobolev Optimization

Our work is in part motivated by Sobolev gradient descent approaches [80, 76, 81, 82, 83, 84, 85, 86, 87, 88] for optimization problems on manifolds, which have been used for segmentation and optical flow problems. These approaches are general in that they apply to non-convex problems, and they are derived by computing the gradient of a cost functional with respect to a Sobolev metric rather than an L^2 metric typically assumed in variational optimization problems. The resulting gradient flows have been demonstrated to yield coarse-to-fine evolutions, where the optimization automatically transitions from coarse to successively finer scale deformations. This makes the optimization robust to local minimizers that plague L^2 gradient descents. We should point out that the Sobolev metric is used outside of optimization problems and have been used extensively in shape analysis (e.g., [89, 90, 91, 92]). While such gradient descents are robust to local minimizers, computing them typically involves an expensive computation of an inverse differential operator at each iteration of the gradient descent. In the case of optimization problems on curves and a very particular form of a Sobolev metric this can be made computationally fast [82], but the idea does not generalize beyond curves. In this work, we aim to obtain robustness properties of Sobolev gradient flows, but without the expensive computation of inverse operators. Our accelerated approach involves averaging the gradient across time in the descent process, rather than an averaging across space in the Sobolev case. Despite our goal of avoiding Sobolev gradients for computational speed, we should mention that our framework is general to allow one to consider accelerated Sobolev gradient descents (although we do not demonstrate it here), where there is averaging in both space and time. This can be accomplished by changing the definition of kinetic energy in our approach. This could be useful in applications where added robustness is needed but speed is not a critical factor.

2.3.6 Variational Active Contours

Geometric partial differential equations have played an important role in image analysis and computer vision for several decades. Applications have ranged from low-level processing operations such using active contours and active surfaces, image registration, and motion estimation via optical flow; to higher level processing such as multiview stereo reconstruction, visual tracking, simultaneous localization and mapping (SLAM), and shape analysis. See, for example, [93, 94, 95] for introductions to PDE methods already established within computer vision within the 1990's, including level set methods [96] already developed in the 1980's for shape propagation. Several such PDE methods have been formulated, using the calculus of variations [18] as gradient descent based optimization problems in functional spaces, including geometric spaces of curves and surfaces. For example, several active contour models are formulated as gradient descent PDE flows of application-specific energy functionals E which relate the unknown contour C to given data measurements. Such energy functionals are chosen to depend only upon the geometric shape of the contour C, not its parameterization. Under these assumptions the first variation of E will have the following form

$$\delta E = -\int_C f\,(\delta C \cdot N)\,ds \tag{2.12}$$

where fN represents a perturbation field along the unit normal N at each contour point and ds denotes the arclength measure. Note that the first variation depends only upon the normal component of a permissible contour perturbation δC. The form of f will depend upon the particular choice of the energy. For example, in the popular Chan-Vese active contour model [97] for image segmentation, f would be expressed by $(I - c_1)^2 - (I - c_2)^2 + \alpha\kappa$ where I denotes the image value at a given contour point, α an arclength penalty weight, κ the curvature at a given contour point, and c_1 and c_2 the means of the image inside and outside the contour respectively. As an alternative example, the geodesic active contour model [98, 99] would correspond to $f = \phi\kappa N - (\nabla\phi \cdot N)N$ where $\phi > 0$ represents a point measurement designed to be small near a boundary of interest and large otherwise. In all cases, though, the gradient descent PDE will the following explicit form.

$$\frac{\partial C}{\partial t} = fN \quad \text{[explicit gradient flow]} \tag{2.13}$$

This class of contour flows, evolving purely in the normal direction, may be implemented implicitly in the level set framework [96] by evolving a function ψ whose zero level set represents the curve C as follows

$$\frac{\partial \psi}{\partial t} = -\hat{f}\|\nabla\psi\| \quad \text{[implicit level set flow]}$$

where $\hat{f}(x,t)$ denotes a spatial extension of $f(s,t)$ to points away from the curve.

2.3.7 Sobolev gradients for Active Contours

The most notorious problem with most active contour and active surface models is that the normal speed function f depends pointwise upon noisy or irregular data measurements, causing immediate fine scale perturbations in the evolving contour which cause it to become very easily attracted to (and trapped within) spurious local minimizers. This often makes the active contour model strongly dependent upon initialization, except for a limited class of convex or poly-convex energy functionals for which numerical schemes can be devised to reach global minimzers reliably. The traditional way to combat this sensitivity is to add strong regularizing terms to the energy functional which penalize fine scale irregularities in the contour shape. Similar problems and regularization strategies are applied in other PDE based optimization applications outside the example of the illustrative active contour being considered here (for example, in Horn and Schunck style optical flow computation [100]).

This energy regularization strategy has two drawbacks. First, most regularizers lead to second order (or higher) diffusion terms in the gradient contour flow, which impose much smaller time step limitations on the numerical discretization of the evolution PDE. Thus, significantly more evolution steps are required, which incurs a heavy computational cost in the minimization process. Second, regularizers, while endowing a level of resistance to noise and spurious structure, impose regularity on the final converged contour as well, making it difficult or impossible to capture features such as sharp corners, or narrow protrusions/inlets in the detected shape. This can lead to unpleasant trade-offs in several applications.

For the illustrative case of active contours, significantly improved robustness in the gradient flow, without additional energy regularization, can be attained by using geometric Sobolev gradients [101, 102, 80, 103] in place of the standard L^2-style gradient used in traditional active contours. We refer to this class of active contours as Sobolev active contours, whose evolution may be described by the following integral-partial-differential equation

$$\frac{\partial C}{\partial t} = (fN) * K \quad \text{[Sobolev gradient flow]} \tag{2.14}$$

Here $*$ denotes convolution in the arclength measure with a smoothing kernel K to invert the linear Sobolev gradient operator. The numerical implementation is not carried out this way, but the expression gives helpful insight into how the Sobolev gradient flow (Equation 2.14) relates to the usual gradient flow (Equation 2.13). Namely, the optimization process (rather than the energy functional itself) is regularized by averaging point-wise gradient forces fN through the kernel K to yield a smoother contour evolution. This does not change the local minimizers of the energy functional, nor does it impose extra regularity at convergence, but it induces a coarse-to-fine evolution behavior [104, 84, 105] in the contour evolution, making it much more resistant to spurious local minima due to noise or other fluctuations in f.

However, while the Sobolev gradient descent method is extremely successful in making an active contour or surface (or other evolving classes of functions) resistant to a large class of unwanted local minimizers, it comes at heavy computation cost. The spatial integration of gradient forces along the evolving front must occur during every time step, and while there are tricks to do this quickly for closed 2D curves [106, 107, 108, 109] there are no convenient alternatives for 3D surfaces, nor for regions (even in 2D) when applying Sobolev gradient flows to other functional objects (images, optical flow, etc.). The linear operator inversion imposes a notable per-iteration cost, which we will instead distribute across iterations in the upcoming accelerated coupled PDE evolution schemes.

2.3.8 Variational Stereo Reconstruction

The problem of multi-view stereo vision is the reconstruction of a 3D object from a collection of calibrated 2D images. The goal of most stereo algorithms is to produce a constrained 3D geometry that accurately corresponds to a set of 2D images [110]. This task can be addressed through a variety of approaches such as point correspondence using feature mapping, dense correspondence, 3D shape methods and deep neural networks [41]. For our research and experiments we focused on shape from silhouette methods, which are a subset of the larger class of 3D shape methods[13] [41].

Silhouette based methods date back to the 1970s and infer the geometry of the underlying structure [111]. In silhouette driven reconstruction the central idea is to estimate the visual hull of the object we wish to reconstruct and to constrain it so that the silhouettes of the reconstructed object correspond to the silhouettes from the calibrated 2D images [112]. One approach for framing the problem is to construct an energy minimization that seeks to directly impose a 3D shape. One of the first endeavours to pose the problem in a variational framework was undertaken by Faugeras and Keriven [113] who proposed minimizing a local matching score. Their work was extended by Yezzi and Soatto [13] who proposed a region based penalty that sought to minimize the re-projection errors of the generated 3D surface. One of the advantages of the latter approach was the ability to impose regularity and smoothness on the reconstructed surface. Their algorithm *stereoscopic-segmentation* was also bidirectional [13] and unlike some of the volumetric methods that preceded them such as space carving [114] could recover portions of the 3D model that were not included in the initial enveloping region. However, one of the limitations of the energy minimization proposed in [13] was that the functional was non-convex and relied on a first-order gradient descent scheme that could become trapped within

[13] The class of 3D shape methods includes a wide variety of approaches such as shape from shading, shape from texture, shape from silhouette and others.

local minimizers. In response to this limitation several convex variational formulations were put forward in the ensuing years [115, 116, 117]. Although these convex cost functionals are less faithful approximations of the reconstruction problem they are in many cases able to outperform the non-convex formulations that preceded them [116]. In recent years there has been renewed interest in non-convex variational stereo reconstruction [118, 119, 16]. In particular our application of PDE acceleration [16] given in subsection 3.6.2 was able to remedy the problem of local minima trapping in stereoscopic-segmentation without the need of excessive regularization.

2.3.9 Variational Optimization Methods

Over the past few decades, many advancements have been made in optimizing the minimization of variational computer vision problems. In this section we very broadly mention some of the prevailing optimization methods. Readers of this dissertation should be aware that there are a variety of methods for solving both convex and non-convex variational computer vision problems. Convex optimization generally refers to problems in which both the function and search area are convex [120]. This has several benefits in that there is a single global minimum and that descent schemes do not get trapped in local minimizers. Over the past few decades there has been a steady progression in convex optimization methods for tackling image processing problems. These include prominent techniques such as Primal Dual, Split Bregman and ADMM [7, 121, 9].

One of the caveats of limiting optimization to convex problems however is that the problems must be engineered with approximations that lend to the usage of a convex energy function. This is straightforward in flat space cases such as denoising or deconvolution but is significantly harder in problems such as active contours or stereo-reconstruction [115]. And while these formulations can be globally minizable in regards to the cost function, they can also sacrifice problem accuracy that would otherwise be maintained by a non-convex framing of the problem. These methods can also be non-trivial to implement and can require significant background in linearization and model formulation.

And while there are several methods such as gradient descent, conjugate gradient, Sobolev gradients and others for attacking non-convex minimizations they are typically slow and can also get trapped in local minimizers in the energy space [101, 102, 80, 103]. And although increasing the amount of regularity can improve performance it does not always suffice. However, using PDE acceleration we demonstrate that not only do we achieve comparable performance to methods such as ADMM and Primal Dual in convex variational problems, but that our framework is also highly applicable to non-convex optimization. We also demonstrate that PDE acceleration is straightforward to implement and show that any variational problem that has been formulated with gradient descent can now be modified to use PDE acceleration.

CHAPTER 3
PDE ACCELERATION

3.1 Introducing the PDE Acceleration Framework

We now develop a general strategy, based on a generalization of the Euclidean case of Wibisono, Wilson, and Jordan's formulation [32] for extending accelerated optimization into the PDE framework. While our approach will be motivated by the variational ODE framework formulated around the Bregman divergence in [32], we will have to address several mathematical, numerical, and computational considerations which do not need to be addressed in finite dimensions.

For example, the evolving parameter vector in finite dimensional optimization can naturally be interpreted as a single moving particle in $\mathbf{R}^n$ with a constant mass which, in accelerated optimization schemes, gains momentum during its evolution. Since the mass is constant and fixed to a single particle, there is no need to explicitly model it. When evolving a continuous curve, surface, region, or function, however, the notion of accumulated momentum during the acceleration process is much more flexible, as the corresponding conceptual mass can be distributed across the entire domain. The outcome of this formulation will be a coupled system of first-order PDE's, which govern the simultaneous evolution of the continuous unknown (curves in the case considered here) and its velocity.

In addition, as pointed out from the onset, the numerical discretization of accelerated PDE models will also differ greatly from existing momentum based gradient descent schemes in finite dimensions. Spatial and temporal steps sizes will be determined based on CFL stability conditions for finite difference approximations of the PDE's and viscosity solution schemes will be required in most cases to propagate through shocks and rarefactions that may occur during the evolution of a continuous front. This is part of the reason we replace the more general Bregman-Lagrangian in [32] with the simpler time-explicit *generalized action* (Equation 2.3), together with the additional benefit that such a choice allows us to work directly with the continuum velocity of the evolving entity (or other generalizations that are easily defined within the tangent space of its relevant manifold) rather than finite displacements utilized by the Bregman divergence (Equation 2.2).

Just as in [32], the energy functional E to be optimized over the continuous infinite dimensional unknown (whether it be a function, a curve, a surface, or a diffeomorphic mapping) will represent the potential energy term $\mathbf{U}$ in the time-explicit *generalized action* (Equation 2.3). Next, a customized kinetic energy term $\mathbf{T}$ will be formulated to incorporate the dynamics of the evolving estimate during the minimization process. Note that just as the evolution time t would represent an artificial time parameter for a continuous gradient descent process, the kinetic energy term will be linked to artificial dynamics incorporated into the accelerated optimization process. As such, the accelerated optimization dynamics can be designed completely independently of any potential physical dynamics in cases where the unknown might be connected with the motion of real objects. Several different strategies can be explored, depending upon the geometry of the specific optimization problem, for defining kinetic energy terms, including various approaches for attributing artificial mass (both its distribution and its flow) to the actual unknown of interest in order to boost the robustness and speed of the optimization process.

Once the kinetic energy term has been formulated, the accelerated evolution will obtained (prior to discretization) using the Calculus of Variations [18] as the Euler-Lagrange equation of the following time-explicit *generalized action integral*

$$\int \frac{t^{k+1}}{k} \Big(\mathbf{T} - \lambda k^2 t^{k-2} \mathbf{U} \Big) \, dt \tag{3.1}$$

In the simple $k = 2$ case, the main difference between the resulting evolution equations versus the classical Principle of Least Action equations of motion (without the time explicit terms in the Lagrangian) is an additional friction-style term whose coefficient of friction decreases inversely proportional to time. This additional term, however, is crucial to the accelerated minimization scheme. Without such a frictional term, the Hamiltonian of the system (the total energy $\mathbf{T} + \mathbf{U}$), would be conserved, and the associated dynamical evolution would never converge to a stationary point. Friction guarantees a monotonic dissipation of energy, allowing the evolution to converge to a state of zero kinetic energy and locally minimal potential energy (the optimization objective).

This yields a natural physical interpretation of accelerated gradient optimization in terms of a mass rolling down a potentially complicated terrain by the pull of gravity (Figure 3.1). In gradient descent, its mass is irrelevant, and the ball always rolls downward by gravity (the gradient). As such the gradient directly regulates its velocity. In the accelerated case, gravity regulates its acceleration. Friction can be used to interpolate these behaviors, with gradient descent representing the infinite frictional limit as pointed out in [32].

Acceleration comes with two advantages. First, whenever the gradient is very shallow (the energy functional is nearly flat), acceleration allows the ball to accumulate velocity as it moves so long as the gradient direction is self reinforcing. As such, the ball approaches a minimum more quickly. Second, the velocity cannot abruptly change near a shallow minimum as in gradient descent. Its mass gives it momentum, and even if the acceleration direction switches in the vicinity of a shallow minimum, the accumulated momentum still moves it forward for a certain amount of time, allowing the optimization process to *look ahead* for a potentially deeper minimizer.

A brief recounting of the flat linear function case[1] that is addressed in [14] and ap-

[1] A far more rigorous analysis and derivation of the flat space case can be found in [14, 122]. We would refer the reader to [34] for an analysis of the optimal damping coefficient.

plied in subsection 3.3.3 is provided here. Consider the calculus of variations problem

$$\underset{u}{min}\ E(u) = \int_{\Omega} \Phi(x, \nabla u) + \psi(x, u)\, dx \tag{3.2}$$

where the minimization of the Euler-Lagrange equation is satisfied by

$$\nabla E(u)\ = \psi(x, u) - div(\nabla \Phi(x, \nabla u)) = 0. \tag{3.3}$$

Where $\Phi = \Phi(x, p)$, $\nabla \Phi = \nabla_p \Phi$ and $\psi = \psi(x, u)$. Note that the gradient $\nabla E(u)$ satisfies $\frac{d}{d\epsilon}\Big|_{\epsilon=0} E(u + \epsilon v) = \int_{\Omega} (\nabla E[u]\, v)\, dx\ \forall$ v smooth. This is denoted as the L^2 gradient. We define the action integral as follows:

$$J[u] = \int_{t_0}^{t_1} k(t) \left(\frac{1}{2} \int_{\Omega} \rho u_t^2\, dx - b(t)E(u) \right)\, dt. \tag{3.4}$$

where $k(t)$ and $b(t)$ are time-dependent weights. Here $p(x)$ represents the mass density and $\frac{1}{2} \int \rho u_t^2 dx$ and $E(u)$ are the kinetic and potential components of the energy respectively. Recall that the PDE accelerated descent equations are the equations of motion in a Lagrangian framing corresponding to the action integral J. To derive the equations of motion we take the variation on J:

$$0 = \frac{d}{d\epsilon}\Big|_{\epsilon=0} J[u + \epsilon v] = \int_{t_0}^{t_1} \int_{\Omega} k(t)\rho u_t - k(t)b(t)\nabla E[u]\, v\, dx \tag{3.5}$$

Assuming smooth v and compact support for $\Omega\ \times\ (t_o, t_1)$ and integrating by parts about t yields:

$$0 = \int_{t_0}^{t_1} \int_{\Omega} \left[-\frac{\partial}{\partial t} k(t)\rho u_t - k(t)b(t)\nabla E[u] \right] v\, dx \tag{3.6}$$

From a simple rearrangement of the terms to the left-hand side we obtain the

PDE accelerated descent equations:

$$\frac{\partial}{\partial t}(k(t)\rho u_t) = -k(t)b(t)\nabla E[u] \tag{3.7}$$

Defining $a = k'(t)/k(t)$ allows us to rewrite the descent equations as:

$$u_{tt} + au_t = -b(t)\rho(t)\nabla E(u) \tag{3.8}$$

Although we have derived the general form of the accelerated descent equation the benefits of this work are not immediately evident. What we will find in the derivations of the CFL conditions from the discrete equations of our PDE is that we are able to realize improved time steps when compared to gradient descent. The gradient descent PDE $u_t = -\nabla E[u]$ is a diffusion equation which requires a time step of $\Delta t \sim \Delta x^2$ for stability. PDE acceleration on the other hand is a wave equation which allows much larger time steps on the order of $\Delta t \sim \Delta x$ [121]. Thus our application of acceleration can also be viewed as a realization of a relaxed CFL condition. Furthermore the fundamental work of [34] proves both energy monotonicity and a linear convergence rate for the accelerated descent in Equation 3.8. This is stated from [34, 14] as follows:

Lemma 1 *(Energy monotonicity [34]) Assume a(t), b(t) ≥ 0 and let u satisfy Equation 3.8. Suppose either* $u(x,t) = g(x)$ *or* $\nabla\Phi(x, \nabla u)\cdot n = 0$ *on* $\delta\Omega$ *Then*

$$\frac{\partial}{\partial t}(K[u] + b(t)E[u]) = -2a(t)K[u] + b'(t)E[u] \tag{3.9}$$

where $K[u] = \int_\Omega \rho u_t^2 dx$*. In particular, total energy is always decreasing provided* $b'(t) \leq 0$ *and* $E[u] \geq 0$*.*

Theorem 1 *(Convergence rate [34]) Let u satisfy Equation 3.8 and let* u^* *be a solution of* $\nabla E[u^*] = 0$ *in* Ω*. Assume* Φ *is uniformly convex in* ∇u*,* Ψ *is convex and* Ψ_{zz}

is bounded above, $u = u^*$ *on* $\partial\Omega$*,* $a(t) = a > 0$ *is constant, and* $b(t) \equiv 1$ *and* $p \equiv 1$*. Then there exists* $C, \beta > 0$ *such that*

$$||u - u^*||^2_{H^1(\Omega)} \leq Cexp(-\beta t) \tag{3.10}$$

The same convergence rate analysis holds for gradient descent $u_t = -\nabla E[u]$ if the conditions on E are satisfied. Another necessary step in the development of this work is the selection of an optimal damping coefficient $a > 0$. Consider the linear PDE acceleration equation from [34, 14]:

$$u_{tt} + au_t + Lu + \lambda u = f \ in \ \Omega \times (0, \infty) \tag{3.11}$$

where L is a linear second-order elliptic operator. The Fourier analysis from [34] gives the optimal choice of the damping parameter as:

$$a = 2\sqrt{\lambda_1 + \lambda} \tag{3.12}$$

where λ_1 is the first non-trivial eigenvalue[2] of L for either the Dirichlet or Neumann boundary conditions. Here the optimal convergence rate is given as:

$$|u(x,t) - u^*(x)| \leq Cexp(-at). \tag{3.13}$$

Note that the case of $\lambda > 0$ corresponds to the presence of a fidelity term in the image processing problem. The fidelity term drives and accelerates the PDE meanwhile the damping coefficient slows the descent and allows for an optimal convergence rate. This algorithm is quite successful for problems such as TV restoration where the problem is not strongly convex but has a fidelity term [14].

[2] In the case where L is degenerate elliptic, $\lambda_1 = 0$. This roughly corresponds to a non-strongly convex optimization problem.

For the later sections (chapter 3) we consider the case of unit density ρ and unit energy scaled b for Equation 3.8 to give the form of the equation used in our experiments:

$$u_{tt} = -au_t - \nabla E(u) \tag{3.14}$$

In the following sections we provide an extensive discussion of discretization strategies and PDE stability analysis. After this we apply the PDE acceleration framework to image denoising, deconvolution, compressed sensing, as a general optimizer for the Poisson equation, optical flow, active contours, and active surfaces.

3.2 Discretization Schemes

In the course of this research we have done a number of experiments and applications of the accelerated PDE framework derived in [14, 34, 16] to build towards the goal of accelerating variational image processing models using PDE acceleration. The accelerated gradient PDE typically has the form of a damped wave equation:

$$u_{tt} = -au_t - \nabla E(u) \tag{3.15}$$

where u_t and u_{tt} are the first and second order derivatives of the evolving entity in time respectively and $\nabla E(u)$ is the gradient of the energy for the evolving entity. Here our primary intended usage of the term “acceleration” refers to the physical interpretation of acceleration as the second time derivative of the evolving entity. Here the au_t term with damping coefficient a acts to dissipate the kinetic energy [14]. Our secondary usage of the term acceleration applies to a special class of variational problems where a less restrictive CFL condition allows coarser time sampling of the discretized PDE’s. Here an improvement in the CFL conditions for these PDE’s is realized numerically by much larger step sizes for both their explicit and semi-implicit discretizations. Thereby this allows another interpretation of the term “acceleration” as a raw computational speed-up, as shown conceptually in Figure 3.1.

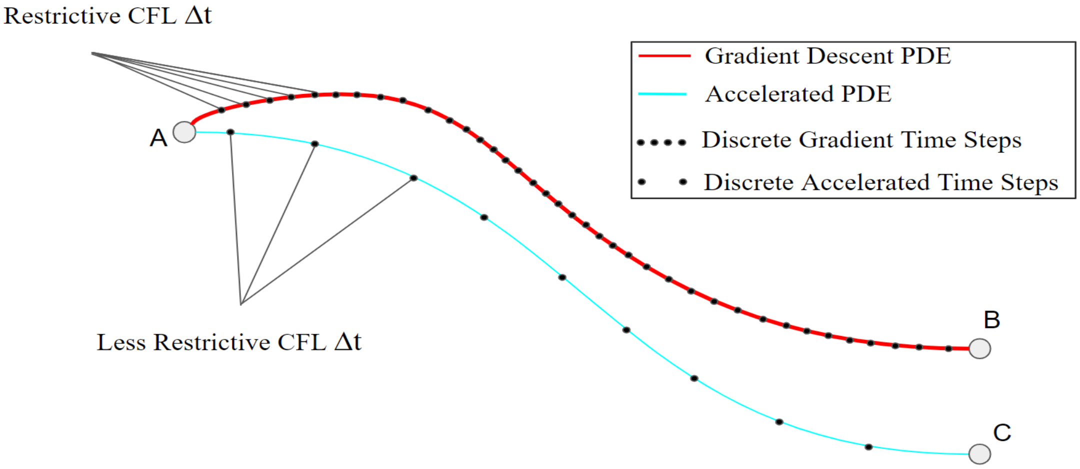


Figure 3.1: Illustration of Gradient descent PDE vs Accelerated Gradient PDE: Gradient descent PDEs and the corresponding accelerated gradient PDEs generally follow different paths. Further, accelerated PDEs lead to discretization schemes with less restrictive CFL conditions corresponding to larger discrete steps than gradient descent PDE’s, leading to faster convergence in the convex case. Note in the case of strictly convex problems B = C.

However, these advantages in improved CFL conditions are not immediately obvious. From the continuous form of the accelerated gradient PDE arises the question of how to best discretize the PDE. To answer this we derive several discretization schemes along with their accompanying CFL conditions.

3.2.1 Explicit Gradient Descent Scheme

To introduce the required mathematical machinery we first begin with an explicit forward Euler discretiziation of the continuous gradient descent PDE. To begin we first write the general form of the first order elliptic equation:

$$u_t = -\nabla E[u] \tag{3.16}$$

Starting with the forward difference in time we expand the left-hand side to obtain:

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = -\nabla E \tag{3.17}$$

From this we can write our update equations:

$$\begin{aligned} \Delta u^n(x) &= -\Delta t \nabla E^n \\ u^{n+1}(x) &= u^n(x) + \Delta u^n(x) \end{aligned} \tag{3.18}$$

In this case $u^n(x)$ denotes the nth update of $u(x,t)$ and Δu^n is $u(x, n\Delta t + \Delta t) - u(x, n\Delta t)$ for a particular n and $u^{n+1}(x) \doteq u(x,(n+1)\Delta t)$. Here the nth iterate of the gradient $\nabla E^n(x)$ corresponds to $\nabla E(x, n\Delta t)$ which is the discrete approximation computed at iteration n. To perform a stability analysis we now consider the required upper bound on the time step Δt. This can be found using a routine CFL analysis for our particular discretization. The typical approach is to perform a Von Neumann analysis and linearization of ∇E^n [11]. We begin by taking a discrete Fourier transform of Equation 3.18.

$$U^{n+1}(\omega) - U^n(\omega) \approx -\Delta t[z(\omega)U^n(\omega)] \tag{3.19}$$

To reach the structure of Equation 3.19 we rely on an explicit computation of ∇E^n using only values of u^n. In this case the linearization of ∇E^n consists only of a combination of u^n values with Discrete Fourier Transform (DFT) of form $z(\omega)U^n(\omega)$. Here $U^n(\omega)$ denotes the DFT of u^n. And $z(\omega)$ is referred to as the gradient amplifier, which is defined as:

$$z(\omega) = \frac{DFT(L^n)}{DFT(u^n)} \tag{3.20}$$

where L^n is the linearized homogeneous component of ∇E^n. From Equation 3.19

we can then write the following update equation:

$$U^{n+1}(\omega) \approx \underbrace{(1 - \Delta t z(\omega))}_{\xi(\omega)} U^n(\omega) \tag{3.21}$$

Here we have an guarantee of stability as long as the update amplification factor $\xi(\omega)$ does not have a complex magnitude that exceeds 1. This condition can be easily expressed as:

$$\xi(\omega)\xi^*(\omega) = (1 - \Delta t z(\omega))(1 - \Delta t z^*(\omega)) \leq 1 \tag{3.22}$$

which then allows us to write our time step restriction.

$$\Delta t \leq \frac{z(\omega) + z^*(\omega)}{z(\omega)z^*(\omega)} = \frac{1}{z(\omega)} + \frac{1}{z^*(\omega)} = 2R\left(\frac{1}{z(\omega)}\right) \tag{3.23}$$

In the special-case of elliptic operators which are common to regularized optimization in the domain of image processing the gradient amplifier term is both real and non-negative: $z(\omega) \geq 0$. This allows us to simplify the stability constraint as follows:

$$\Delta t \leq \frac{2}{z_{max}} \tag{3.24}$$

Here z_{max} is the $max_\omega\, z(\omega)$.

3.2.2 First Order Accelerated Scheme

We now present the explicit discretizations of the accelerated PDE (Equation 3.15). However to undertake the required analysis we will require an additional tool, the Root Amplitude Lemma [14]. The lemma is restated here for the benefit of the reader.

Lemma 2 ***Root Amplitude Lemma*** *Given a quadratic equation* $A\xi^2 + B\xi + C = 0$ *with real coefficients* $(A \neq 0)$*, its roots will satisfy* $|\xi| \leq 1$ *if and only if* $\frac{|B|}{|A|} - 1 \leq \frac{C}{A} \leq 1$

(or equivalently $A > C$ and $A + C > |B|$ for positive A [3]).

To write the forward difference scheme for Equation 3.15 we begin by using a second order central difference for u_{tt} and a forward difference for u_t from equation Equation 3.15.

$$\frac{u(x, t + \Delta t) - 2u(x, t) + u(x, t - \Delta t)}{\Delta t^2} + a\frac{u(x, t + \Delta t) - u(x, t)}{\Delta t} = -\nabla E(x, t) \quad (3.25)$$

Using the same notation as in the gradient descent case we rewrite this as:

$$\frac{u^{n+1} - 2u^n + u^{n-1}}{\Delta t^2} + a\frac{u^{n+1} - u^n}{\Delta t} = -\nabla E \quad (3.26)$$

With some minor algebraic manipulation we are able to obtain the corresponding update equation:

$$u^{n+1}(x) = \frac{(2 + a\Delta t)u^n - u^{n-1} - \Delta t^2 \nabla E^n}{1 + a\Delta t} \quad (3.27)$$

We then perform a DFT to find the frequency domain form of the update equation:

$$U^{n+1}(\omega) = \frac{(2 + a\Delta t - \Delta t^2 z(\omega))U^n - U^{n-1}}{1 + a\Delta t} \quad (3.28)$$

where $z(\omega)$ again denotes the gradient amplifier [123]. We can then substitute $U^{n\pm m} = \xi^{\pm m} U^n$, where $\xi(\omega)$ is again given as the overall update amplification factor. The remaining math is given as follows: first multiply the left hand side by the denominator, move all terms to the left and substitute the remaining terms. This gives the equation:

$$(1 + a\Delta t)\xi^2 + (\Delta t^2 z(\omega) - 2 - a\Delta t)\xi + 1 = 0 \quad (3.29)$$

[3] A proof of this lemma can be found in [14].

where $U^{n+1} \mapsto \xi^2$, $U^n \mapsto \xi$ and $\nabla E^n \mapsto z(\omega)U^n \mapsto z(\omega)\xi$. From this we assign coefficients for a quadratic equation:

$$\underbrace{(1 + a\Delta t)}_{A}\xi^2 + \underbrace{(\Delta t^2 z(\omega) - 2 - a\Delta t)}_{B}\xi + \underbrace{1}_{C} = 0 \tag{3.30}$$

where $\xi = \xi(\omega)$ and move to apply the Root Amplitude Lemma. Since $1 + a\Delta t > 1 \; \forall \; a, \Delta t > 0$ we satisfy the first condition of the root amplitude lemma for $A \geq C \; \forall \; A > 0$. From this we address the second condition: $A + C \geq |B|$, with the assumption that $z(\omega)$ is real. To satisfy the CFL condition we must ensure that $|\xi(\omega)| \leq 1$. This translates to the condition that $(1+a\Delta t)+1 \geq |\Delta t^2 z(\omega) - (2+a\Delta t)|$ which equates to $0 \leq \Delta t^2 z(\omega) \leq 2(2 + a\Delta t)$. In the case of $z(\omega) \geq 0$ we satisfy the left-hand inequality for all ω which leaves $z(\omega)\Delta t^2 - 2a\Delta t - 4 \leq 0$. Using the quadratic equation and substituting z_{max} as the worst possible case of $z(\omega)$ yields:

$$\Delta t \leq \sqrt{\frac{4}{z_{max}} + \left(\frac{a}{z_{max}}\right)^2} + \frac{a}{z_{max}} \tag{3.31}$$

This scheme allows for larger time steps than gradient descent. However it comes with a drawback as increasing the time step requires a larger damping coefficient. While this can allow for bigger jumps increasing the damping beyond the optimal choice will actually result in a decreased convergence rate[4]. In [14] we derive an optimal damping coefficient for TV denoising and show that the method is slower when it is either under or over damped. We also go on to show that by raising the damping even further the descent algorithm reverts to a sub-gradient method that is decidedly slower than gradient descent.

[4] The first order scheme, although it has a larger time step actually results in a smaller increment in the update equation. The nuisances of this are more fully discussed in [14] and are left as an exercise to the reader.

3.2.3 Second Order Accelerated Scheme

The second order accelerated scheme from [14] can be derived as follows. We start by writing Equation 3.15 in the form of a sum of two central differences one for the velocity term and another for the acceleration.

$$\frac{u(x,t+\Delta t)-2u(x,t)+u(x,t-\Delta t)}{\Delta t^2}+a\frac{u(x,t+\Delta t)-u(x,t-\Delta t)}{2\Delta t}=-\nabla E(x,t) \tag{3.32}$$

with some minor manipulation we can write the update equation. Treating $u(x,t+\Delta t)$ as u^{n+1}, $u(x,t)$ as u^n and $u(x,t-\Delta t)$ as u^{n-1} yields the update of u:

$$u^{n+1}(x)=\frac{2u^n-(1-\frac{a\Delta t}{2})u^{n-1}-\Delta t^2\nabla E^n}{1+\frac{a\Delta t}{2}} \tag{3.33}$$

To obtain the stability condition we apply the DFT to the linearized homogeneous portion of the update scheme obtaining:

$$U^{n+1}(\omega)=\frac{(2-\Delta t^2 z(\omega))U^n-(1-\frac{a\Delta t}{2})U^{n-1}}{1+\frac{a\Delta t}{2}} \tag{3.34}$$

Using the same manner of substitution as in the first order case we obtain another quadratic equation:

$$\underbrace{\left(1+\frac{a\Delta t}{2}\right)}_{A}\xi^2+\underbrace{\left(\Delta t^2 z(\omega)-2\right)}_{B}\xi+\underbrace{\left(1-\frac{a\Delta t}{2}\right)}_{C}=0 \tag{3.35}$$

with coefficients A, B and C. From this we apply the Root Amplitude Lemma to check the stability condition $|\xi(\omega)|\leq 1$. The first condition $A\geq C$ for positive A is satisfied as $1+\frac{a\Delta t}{2}\geq 1-\frac{a\Delta t}{2}$ for all positive a and Δt. Next we consider the second condition: $A+C\geq|B|$ which yields the stability condition $2\geq|2-\Delta t^2 z(\omega)|$. This can be rewritten as $0\leq\Delta t^2 z(\omega)\leq 4$; in the case of $z(\omega)\geq 0$ we automatically satisfy the left hand side leaving us with:

$$\Delta t \leq \frac{2}{\sqrt{z_{max}}} \tag{3.36}$$

We summarize the explicit schemes[5] and their stability conditions below:

$$\text{Gradient Descent: } \Delta u^n = -\Delta t \nabla E^n,\ \Delta t \leq \frac{2}{z_{max}} \tag{3.37}$$

$$\text{1-Order: } \Delta u^n = \frac{1}{1+a\Delta t} \cdot \Delta u^{n-1} - \frac{\Delta t^2}{1+a\Delta t} \cdot \nabla E^n,\ \Delta t \leq \sqrt{\frac{4}{z_{max}} + \left(\frac{a}{z_{max}}\right)^2} + \frac{a}{z_{max}} \tag{3.38}$$

$$\text{2-Order: } \Delta u^n = \frac{2-a\Delta t}{2+a\Delta t} \cdot \Delta u^{n-1} - \frac{2\Delta t^2}{2+a\Delta t} \cdot \nabla E^n,\ \Delta t \leq \frac{2}{\sqrt{z_{max}}} \tag{3.39}$$

These single increment descent schemes are obtained through algebraic manipulation of the preceding update formulations: Equation 3.18, Equation 3.27, and Equation 3.33. The new accelerated gradient PDEs provide two principal advantages over standard gradient descent. The first is that for special classes of variational problems we are able to write far less restrictive CFL conditions. This allows for much larger time steps and coarser sampling of the discretized PDE. The second advantage is that by incorporating a momentum term we are able to bypass many of the local minimizers in our problem space and converge to much better optima; see Figure 3.1 for a more complete conceptualization of this.

3.2.4 Backward Difference

An interesting exercise is to use backward differences in time for Equation 3.15. To do this we begin with the same second order central difference on u_{tt} but then take a

[5] As an aside, a number of discretization strategies and schemes were derived and evaluated in [14]. We focus on the second order scheme here as it was what we used for our experiments against convex optimizers.

backwards difference on u_t:

$$\frac{u(x,t+\Delta t)-2u(x,t)+u(x,t-\Delta t)}{\Delta t^2}+a\frac{u(x,t)-u(x,t-\Delta t)}{\Delta t}=-\nabla E(x,t) \quad (3.40)$$

From this we write the corresponding update equation:

$$u^{n+1}(x)=(2-a\Delta t)u^n-(1-a\Delta t)u^{n-1}-\Delta t^2\nabla E^n \quad (3.41)$$

A similar analysis yields the quadratic equation:

$$\xi^2(\omega)-\Big(2-a\Delta t-\Delta t^2 z(\omega)\Big)\xi(\omega)+(1-a\Delta t)=0 \quad (3.42)$$

for the amplification factor $\xi(\omega)$. Applying the Root Amplitude Lemma we see that the first condition $A \geq C$ of the lemma is satisfied as $(1 \geq 1-a\Delta t)$ for all positive values of a, Δt. The second condition is to satisfy $A+C \geq |B|$, with the assumption real $z(\omega) \geq 0$. With a simple rearrangement of the quadratic coefficients we find the desired inequality: $\Delta t^2 z(\omega)+2a\Delta t-4 \leq 0$. Solving for the roots with the max value z_{max} substituted for $z(\omega)$ yields:

$$\Delta t \leq \sqrt{\frac{4}{z_{max}}+(\frac{a}{z_{max}})^2}-\frac{a}{z_{max}} \quad (3.43)$$

Notice that the CFL conditions from the prior forward and central difference schemes, Equation 3.27 and Equation 3.33 respectively are necessary but not sufficient. Furthermore we observe that the constraint on the time step becomes increasingly restrictive as the damping coefficient a increases. This is in direct opposition to the forward scheme and makes it impossible to consider a damping-independent stability constraint.

3.2.5 Semi-Implicit Schemes

It is feasible to employ semi-implicit Euler style discretizations of Equation 3.15 to obtain systems which more closely resemble classical two-part Nesterov recursion [40]. This can be done with any of the fully explicit schemes Equation 3.18, Equation 3.27, or Equation 3.33 by replacing the explicit discretization of ∇E^n of the gradient with a "predicted estimate" $\nabla \hat{E^{n+1}}$ of its implicit discretization $\nabla E^{n+1} \approx \nabla E(v^n)$. The partial update v^n is obtained before-hand via the fully explicit update without the gradient term (i.e., by treating ∇E^n as if it were zero). Using this strategy with the second order in time scheme Equation 3.33 yields the two-step update:

$$v^n = u^n + \frac{2 - a\Delta t}{2 + a\Delta t} \Delta u^{n-1} \tag{3.44}$$

$$u^{n+1} = v^n - \frac{2\Delta t^2}{2 + a\Delta t} \nabla E(v^n) \tag{3.45}$$

Notice that the first and second steps, separately both have a fully explicit structure. Now we employ Von Neumann analysis to analyze the stability of this scheme according to the following update relationships between the DFT sequences U^n, V^n and U^{n+1} (frequency transforms of u^n,v^n, and u^{n+1}, respectively). Again we treat $z(\omega)$ as the gradient amplifier associated with the linearization of ∇E^n (and therefore we can also extend this to the linearization of $\nabla \hat{E^{n+1}}$). From the the DFT we obtain the following:

$$V^n = \frac{4}{2 + a\Delta t} U^n - \frac{2 - a\Delta t}{2 + a\Delta t} U^{n-1} \tag{3.46}$$

$$U^{n+1} = \left(1 - \frac{2\Delta t^2}{2 + a\Delta t} z(\omega)\right) V^n \tag{3.47}$$

Now we substitute the first equation into the second and follow this by the familiar substitutions $U^{n\pm m} = \xi^{\pm m} U^n$ to obtain our quadratic equation:

$$(2+\alpha\Delta t)^2\xi^2-4\Bigg(2+\alpha\Delta t-2\Delta t^2 z(\omega)\Bigg)\xi+(2-a\Delta t)\Bigg(2+a\Delta t-2\Delta t^2 z(\omega)\Bigg) = 0 \quad (3.48)$$

for the overall combined update amplification factor $\xi(\omega)$. Now we apply the Root Amplitude Lemma to check the stability criteria for $|\xi(\omega)| \leq 1$.

First we test $A \geq C$ for positive A. From the coefficients in our quadratic equation this is expressed as $az\Delta t^2 - (a^2+2z)\Delta t - 2a \leq 0$. This condition will be satisfied between its positive and negative roots. Restricting our interest to only positive values of Δt therefore yields the constraint

$$\Delta t \leq \underbrace{\frac{a^2+2z+\sqrt{(a^2+2z)^2+8a^2z}}{2az}}_{g(a,z)>0} \tag{3.49}$$

where

$$\frac{\partial g}{\partial a} = \underbrace{(a^2-2z)}_{toggles}\underbrace{\frac{a^2+2z+\sqrt{(a^2+2z)^2+8a^2z}}{2a^2z\sqrt{(a^2+2z)^2+8a^2z}}}_{always positive} \tag{3.50}$$

To satisfy this independently of a, we examine the partial derivative of the upper bound g(a,z) with respect to a see that it starts out negative for $a^2 < 2z$ and then turns positive for $a^2 > 2z$. The minimum upper bound is therefore attained when $z(\omega) = z_{max}$ and $a^2 = 2z_{max}$ yielding

$$\Delta t \leq \frac{2+\sqrt{2}}{\sqrt{z_{max}}} \tag{3.51}$$

While this upper bound is more generous than the second order explicit scheme, it only satisfies the first of the two stability conditions in the bounded root lemma. We now proceed to the second condition which will be more restrictive.

The second conditions for the Root Amplitude Lemma: $A + C \geq |B|$ can be expressed as follows:

$$2 + a\Delta t - \Delta t^2 z(\omega) + \frac{1}{2} a\Delta t^3 z(\omega) \geq |2 + a\Delta t - 2\Delta t^2 z(\omega)| \tag{3.52}$$

For small enough time steps, $2 + a\Delta t - 2\Delta t^2 z$ is positive and we can remove the absolute values from the inequality. For larger time steps, $2 + a\Delta t - 2\Delta t^2 z$ becomes negative and the inequality can be rearranged into the following cubic form:

$$az(\omega)\Delta t^3 - 6z(\omega)\Delta t^2 + 4a\Delta t + 8 \geq 0. \tag{3.53}$$

Minimizing on on the left with the case $a = 0$ and $z(\omega) = z_{max}$ yields a stricter and therefore sufficient, stand-alone stability conditions

$$\Delta t \leq \frac{2}{\sqrt{3z_{max}}}. \tag{3.54}$$

Note that this upper bound is smaller, by a factor of $\sqrt{3}$, than the maximum stable time step for the for the fully explicit second order scheme.

3.3 Regularized Inversion Problems

3.3.1 General Case (nonlinear wave equation)

To address the regularized inversion problems presented in this dissertation we first consider a very general class of variational regularized inversion problems in the accelerated PDE framework. In particular, we assume energy functions with the form

$$E(u) = \int_\Omega \underbrace{f\left(|\mathcal{K}u - g|\right)}_{\text{fidelity}} + \underbrace{r(\|\nabla u\|)}_{\text{regularity}} \; dx$$

with $\dot{f}, \dot{r}, \ddot{r} > 0$. The function f is a monotonically increasing penalty on the residual error between data measurements g and a forward in the form of linear operator $\mathcal{K}$ applied to the reconstructed signal u, while r is a monotonically increasing penalty on the gradient of the reconstruction.

The continuum gradient of E has the form

$$\begin{aligned}\nabla E(u) = &\underbrace{\frac{\dot{f}\left(|\mathcal{K}u - g|\right)}{|\mathcal{K}u - g|}}_{\lambda(u,x)>0} \mathcal{K}^* \left(\mathcal{K}u - g\right) \\ &- \underbrace{\frac{\dot{r}(\|\nabla u\|)}{\|\nabla u\|}}_{c(\nabla u)>0} \left(\nabla \cdot \nabla u - u_{\eta\eta}\right) \\ &- \underbrace{\ddot{r}(\|\nabla u\|)}_{d(\nabla u)>0} u_{\eta\eta},\end{aligned}$$

where $\mathcal{K}^*$ denotes the adjoint of the forward operator $\mathcal{K}$, and where $\eta \doteq \frac{\nabla u}{\|\nabla u\|}$ denotes the unit vector along the gradient direction of u. This gives rise to the following class of accelerated flows which take the form of a nonlinear wave equation.

$$\begin{aligned}u_{tt} - \; c(\nabla u) \; (\nabla \cdot \nabla u - u_{\eta\eta}) \; - \; d(\nabla u) \; u_{\eta\eta} + a u_t \\ = \lambda(u,x) \; \mathcal{K}^* \left(g - \mathcal{K}u\right)\end{aligned}$$

If, purely for the sake of understanding stability, we model the short time behavior of any of the presented discrete update schemes in the neighborhood of a particular spatial point x, by treating λ, c, and d as locally constant, and by representing the forward model linear operator $\mathcal{K}$ as a real convolution kernel K with adjoint K^T, then ∇E can be approximated near x by the following linear expression

$$\begin{aligned}\nabla E \approx \lambda_{[x]} \, K_{[x]}^T * \left(K_{[x]} * u^n - g\right) \\ - c_{[x]} \; \left(\nabla \cdot \nabla u^n - u_{\eta\eta}\right) + d_{[x]} \, u_{\eta\eta}\end{aligned} \tag{3.55}$$

where the subscript $[x]$ denotes the local point of spatially constant approximation (rather than a function argument). Assuming a uniform Cartesian grid oriented such that its first basis vector $\vec{e_1} = (1, 0, 0, \ldots)$ aligns with ∇u at our local point x, and that our spatial derivative discretizations become equivalent to central difference (second derivative) approximations with space step Δx in each direction, then we obtain the following local approximation of the gradient amplifier of (Equation 3.55)

$$\begin{aligned} z(x, \omega_1, &\omega_2, \ldots, \omega_N) \\ &\approx \lambda_{[x]} \operatorname{DFT}(K_{[x]}^T) \operatorname{DFT}(K_{[x]}) \\ &\quad + \frac{2}{\Delta x^2}\Big(d_{[x]}\,(1 - \cos \omega_1 \Delta x) \\ &\qquad + c_{[x]} \sum_{k=2}^{N} (1 - \cos \omega_k \Delta x)\Big) \end{aligned} \tag{3.56}$$

Noting that the Fourier transform of the adjoint K^T of a real convolution kernel is always the complex conjugate of the Fourier transform of the kernel K itself, we see that the gradient amplifier is real and positive and we can write the following upper bound as a function frequency ω

$$\max_{\omega} z \leq \lambda_{[x]} \max_{\omega} |\operatorname{DFT}(K_{[x]})|^2 + 4\,\frac{c_{[x]}\,(N-1) + d_{[x]}}{\Delta x^2} \tag{3.57}$$

with equality in cases where the complex amplitude of $\operatorname{DFT}(K)$ is maximal at $\omega = (\pi, \ldots, \pi)$. However, since this upper bound depends on the local point of approximation x, we need to maximize over x as well in order to exploit the CFL formulas presented earlier in terms of $z_{\max}$. Doing so yields the following upper bound for the local gradient amplifier.

$$z_{\max} \leq K_{\max} \lambda_{\max} + 4\,\frac{(N-1)c_{\max} + d_{\max}}{\Delta x^2} \tag{3.58}$$

where $\lambda_{\max} \doteq \max_x \lambda$, $c_{\max} \doteq \max_x c$, $d_{\max} \doteq \max_x d$, and $K_{\max} \doteq \max_{x,\omega} (|\operatorname{DFT}(K)|^2)$.

If we now plug (Equation 3.58) into the time step restriction Equation 3.36 for the fully explicit second order accelerated scheme Equation 3.39, we obtain the following sufficient condition for stability

$$\Delta t \leq \frac{2\Delta x}{\sqrt{K_{\max}\lambda_{\max}\Delta x^2 + 4(N-1)c_{\max} + 4d_{\max}}} \tag{3.59}$$

The corresponding condition for gradient descent is obtained by squaring Δx in the numerator and removing the radical (squaring) the denominator. As such we note three favorable step size trends for PDE acceleration compared to PDE gradient descent. Most notably, when the regularizing coefficients $c_{\max}$ and $d_{\max}$ dominate, stable time step sizes are now directly proportional to spatial step sizes rather than to their squares, making the upper bound linear rather than quadratic in Δx. We see similar gains as well when the kernel K exhibits large amplification at one or more frequencies. In such cases, stable step sizes are inversely proportional to the maximum kernel amplification rather than to its square.

3.3.2 Image Denoising

One of the most straightforward cases to apply PDE acceleration to is the case of a quadratic fidelity without any type of forward model, K. We first consider the ROF cost functional with an L_2 norm [11]:

$$E(u) = \int_\Omega \frac{\lambda}{2}(u-g)^2 + \frac{c}{2}||\nabla u||^2 dx \tag{3.60}$$

for denoising. Here the unique minimizer of $E(u)$ is given as the solution to the elliptic problem [3]. To see this we take the first variation on the cost functional to compute the gradient of the energy. We begin with

$$\lim_{\epsilon \to 0} \frac{E(u+\epsilon v) - E(u)}{\epsilon} \tag{3.61}$$

which expands as

$$\frac{E(u+\epsilon v)-E(u)}{\epsilon}=\frac{\int_\Omega \frac{\lambda}{2}(u+\epsilon v-g)^2+\frac{c}{2}||\nabla(u+\epsilon v)||^2dx-\int_\Omega \frac{\lambda}{2}(u-g)^2+\frac{c}{2}||\nabla(u)||^2dx}{\epsilon} \tag{3.62}$$

To improve readability let us first consider the diffusion term:

$$\frac{\int_\Omega \frac{c}{2}||\nabla(u+\epsilon v)||^2dx-\int_\Omega \frac{c}{2}||\nabla(u)||^2dx}{\epsilon} \tag{3.63}$$

we rewrite this as:

$$\frac{\frac{c}{2}\int_\Omega(||\nabla u||^2+\epsilon^2||\nabla v||^2+2\epsilon\nabla u\nabla v-||\nabla u||^2)dx}{\epsilon} \tag{3.64}$$

simplifying terms and taking the $\lim_{\epsilon\to 0}$ yields:

$$c\int_\Omega(\nabla u\nabla v)dx \tag{3.65}$$

If we assume that our boundary is insulated and that the evolving entity u does not permeate across the boundary we can write the following:

$$n\cdot\nabla u=0,\ \text{on } \partial\Omega \tag{3.66}$$

where n denotes the unit exterior normal of Ω. Using integration by parts and Equation 3.66 we are able to write this as:

$$c\int_\Omega(-\nabla^2uv)dx \tag{3.67}$$

Although v is not free, we can use any constant term to write:

$$c\int_\Omega(-\nabla^2u[v+m])dx \tag{3.68}$$

since we know that $\int_\Omega(-\nabla^2 u[m])dx = 0$. We can write the gradient of the energy of the diffusion term as:

$$-c \cdot \nabla\nabla u \tag{3.69}$$

We now consider the remaining term of the energy gradient:

$$\frac{\int_\Omega \frac{\lambda}{2}(u + \epsilon v - g)^2 dx + \int_\Omega \frac{\lambda}{2}(u - g)^2 dx}{\epsilon} \tag{3.70}$$

we expand the expression:

$$\frac{\lambda}{2}\frac{\int_\Omega[(u^2 + \epsilon^2 v^2 + 2\epsilon vu - 2\epsilon vg - 2ug - g^2) - (u^2 - 2ug - g^2)]dx}{\epsilon} \tag{3.71}$$

simplifying and taking the $\lim_{\epsilon\to 0}$ yields:

$$\lambda \int_\Omega (u - g)v dx \tag{3.72}$$

through a similar manipulation as was done for the diffusion term we can write the gradient of the energy for the quadratic fidelity as:

$$\lambda(u - g) \tag{3.73}$$

and finally we are able to write the gradient of our energy ∇E as:

$$\nabla E = \lambda(u - g) - c\nabla \cdot \nabla u. \tag{3.74}$$

Combing the gradient of the energy with Equation 3.15 allows us to write the accelerated descent PDE in the form of a damped inhomogeneous linear wave equation.

$$u_{tt} - c\nabla \cdot \nabla u + au_t = \lambda(g - u). \tag{3.75}$$

From this we can easily compute the gradient amplifier $z(\omega)$. Using central differences on a uniform N-dimensional Cartesian grid with space step Δx in each direction are used to approximate the spatial derivatives of the Laplacian $\nabla \cdot \nabla$, then

$$z(\omega) = \lambda + \frac{2c}{\Delta x^2} \sum_{k=1}^{N} (1 - cos[\omega_k \Delta x]) \tag{3.76}$$

where $\omega = (\omega_1, ..., \omega_N)$. This allows us to write an upper bound for z_{max}:

$$z_{max} = \lambda + \frac{4N_c}{\Delta x^2} \tag{3.77}$$

With the upper bound on the time step in hand, we have the necessary tools to consider the accelerated second order update equation for the denoising problem:

$$\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t} \cdot \Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t} \bigg(\lambda(u^n - g) - c\nabla\nabla u^n \bigg) \tag{3.78}$$

We now compare PDE acceleration to two popular competing approaches Primal Dual and Split Bregman. In Figure 3.2 we observe an over-diffused restoration of a noisy image. One of the key advantages of the PDE acceleration for this convex problem is the speed of convergence. Although the results for PDE acceleration and Split-Bregman are nearly identical visually PDE acceleration is almost an order of magnitude faster. It is likewise competitive with fast convex optimization methods such as Primal-Dual.

In Figure 3.3 we observe an appropriately diffused restoration of the same noisy image as in Figure 3.2. Again we observe that PDE acceleration is roughly an order

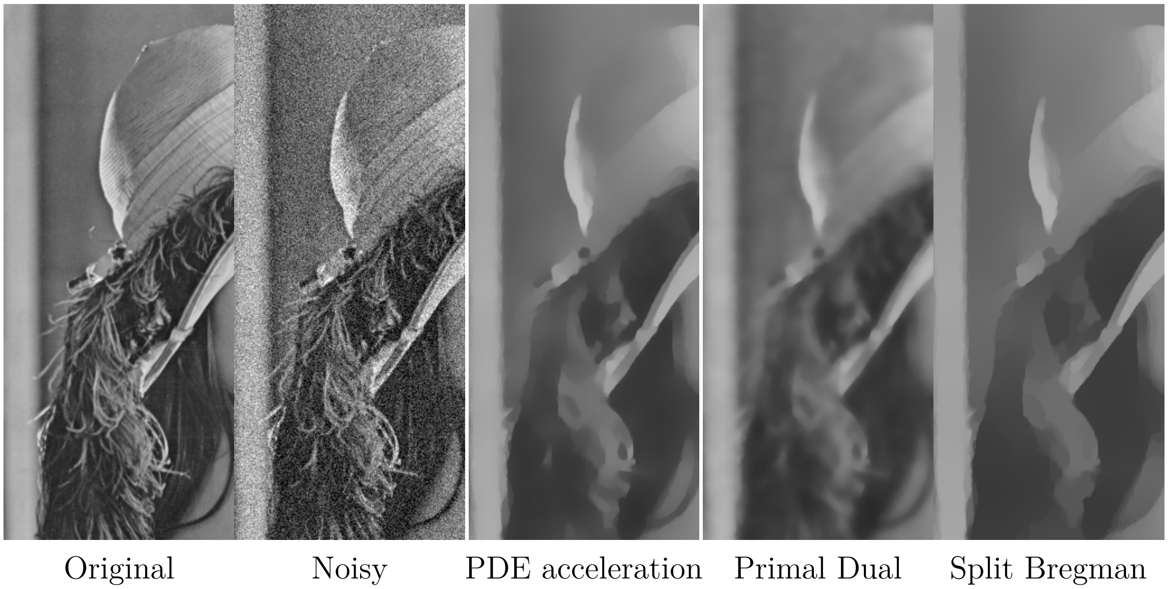


Figure 3.2: Comparison of PDE acceleration, Primal Dual, and Split Bregman for TV restoration of a noisy Lenna image with $\lambda = 1000$. Each algorithm was run for 150 iterations, which took 2.7 seconds for PDE acceleration, 3.3 seconds for Primal Dual, and 28 seconds for Split Bregman.

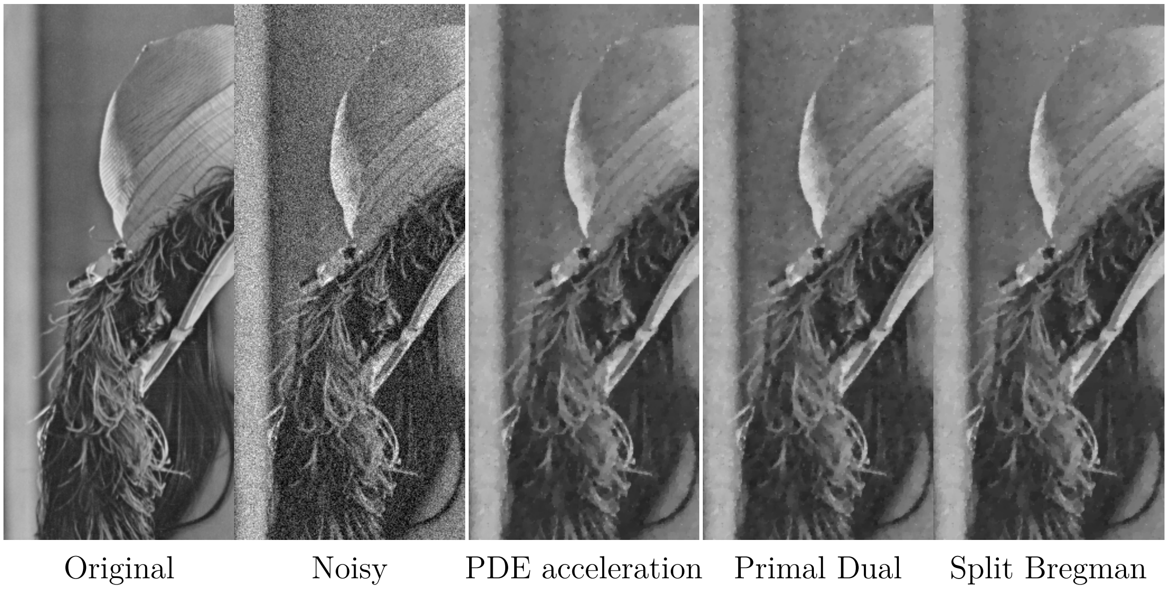


Figure 3.3: Comparison of PDE acceleration, Primal Dual, and Split Bregman for TV restoration of a noisy Lenna image with $\lambda = 7000$. Each algorithm was run for 50 iterations, which took 0.85 seconds for PDE acceleration, 1.12 seconds for Primal Dual, and 10.4 seconds for Split Bregman.

of magnitude faster than the Split Bregman method while still remaining competitive and even outperforming Primal-Dual. Another application of variational denoising is to consider the problem of Beltrami regularization. For this we begin by writing a modified version of the ROF functional with a Beltrami regularizer.

$$E(u) = \int_\Omega \frac{\lambda}{2}(u-g)^2 + \frac{1}{\beta}\sqrt{1 + ||\beta \nabla u||^2} \; d\mathbf{x} \tag{3.79}$$

The Beltrami regularization term interpolates between the TV norm $\int |\nabla u|$ and the L^2 norm $\int |\nabla u|^2$—near edges where ∇u is large, it behaves like the TV norm to preserve edges, and where ∇u is small it behaves like the L^2 norm in order to reduce staircasing. From here we write the expression for the energy gradient:

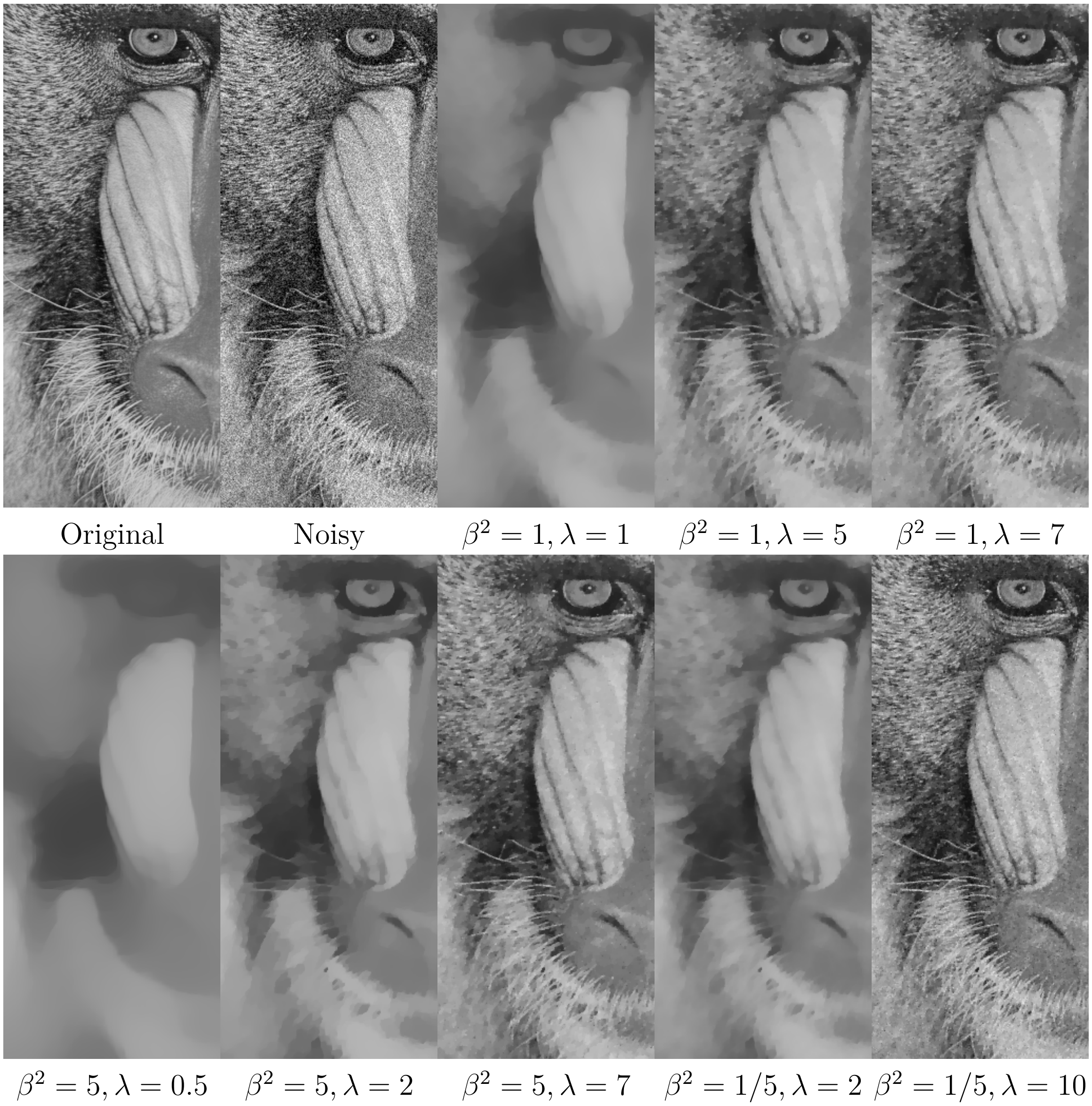
$$\nabla E = \lambda(u-g) - \nabla(\frac{\beta \nabla u}{\sqrt{1 + ||\beta \nabla u||^2}}) \tag{3.80}$$

Plugging the gradient of the energy into Equation 3.39 yields our new update equation:

$$\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t} \cdot \Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t}\left(\lambda(u^n - g) - \nabla(\frac{\beta \nabla u^n}{\sqrt{1 + ||\beta \nabla u^n||^2}})\right) \tag{3.81}$$

Recently, Zosso and Bustin [124] proposed an efficient primal dual projected gradient method for solving Beltrami regularized problems. Figure 3.4 shows the results of applying the PDE accelerated Beltrami regularization to a noisy baboon test image with varying values of λ and β with single-threaded C++ code on a 3.2 GHz Intel processor running Linux.

The corresponding runtimes are given in Table 3.1 and are favorably competitive with the runtimes reported in [124]. Notice the algorithm does slow down somewhat when λ is small and the denoising is heavily regularized, but the difference is far less



Original Noisy $\beta^2 = 1, \lambda = 1$ $\beta^2 = 1, \lambda = 5$ $\beta^2 = 1, \lambda = 7$

$\beta^2 = 5, \lambda = 0.5$ $\beta^2 = 5, \lambda = 2$ $\beta^2 = 5, \lambda = 7$ $\beta^2 = 1/5, \lambda = 2$ $\beta^2 = 1/5, \lambda = 10$

Figure 3.4: Results of Beltrami regularization applied to a noisy baboon image with varying values of λ and β. The units of λ are thousands.

Table 3.1: PDE accelerated Beltrami regularization runtimes on the 512×512 baboon image.

	$\lambda = 1000$		$\lambda = 5000$		$\lambda = 7000$	
	Time	Iterations	Time	Iterations	Time	Iterations
$\beta^2 = 1/5$	0.55s	124	0.27s	60	0.23s	50
$\beta^2 = 1$	0.81s	183	0.38s	85	0.32s	71
$\beta^2 = 5$	1.20s	273	0.54s	122	0.45s	101

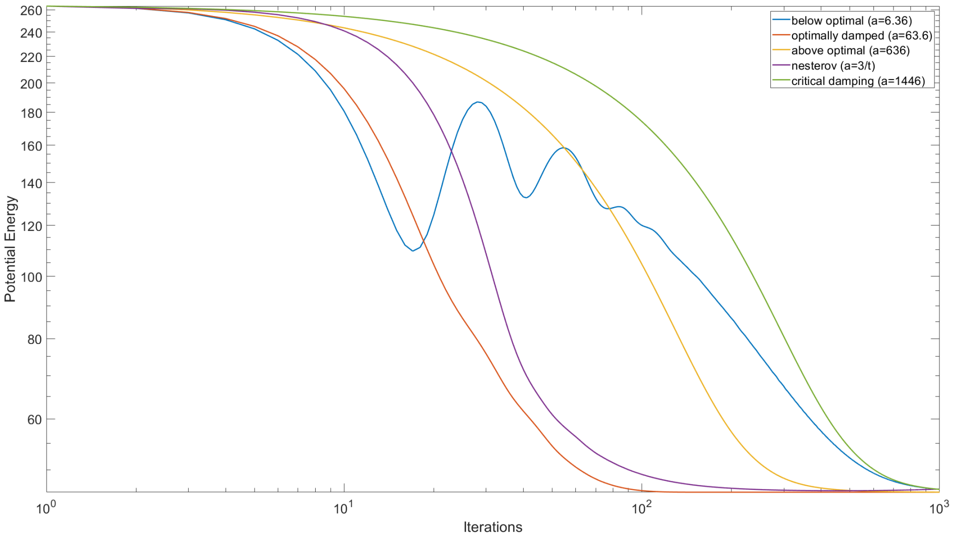


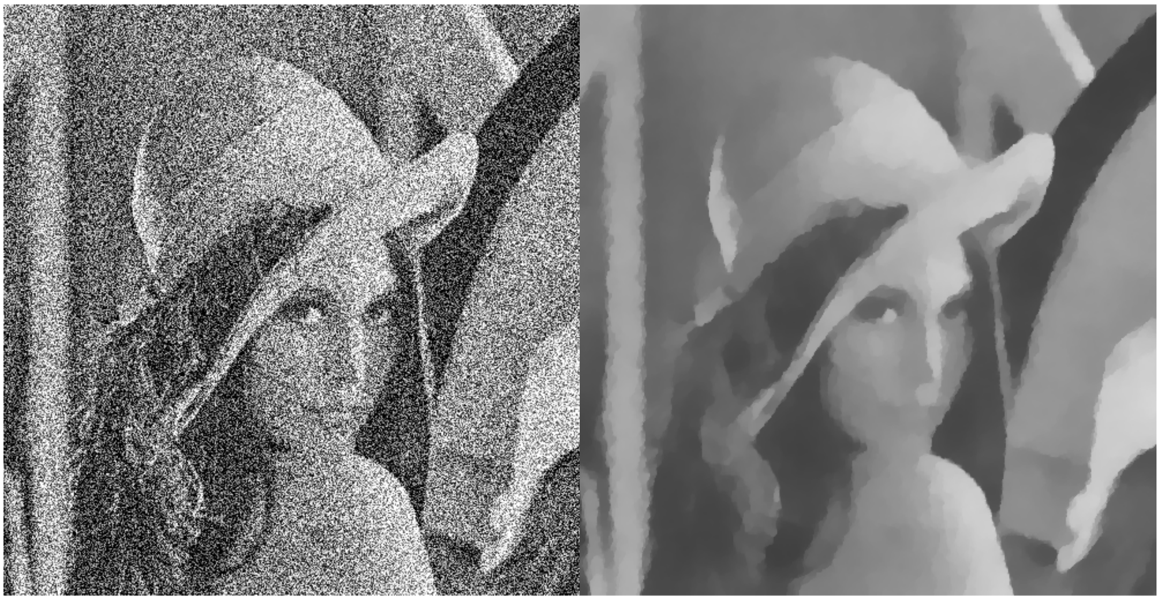
Figure 3.5: Convergence experiment with variable damping for a Beltrami regularizer. Initial condition and converged result are given in Fig Figure 3.6.

Table 3.2: Coefficients and converge times of damping experiments for a 512 x 512 noisy Lena image.

Damping	Below Optimal		Optimal		Above Optimal		Nesterov		Critical	
	a	Iterations	a	Iterations	a	Iterations	a	Iterations	a	Iterations
$\lambda = 1000$, $\beta^2 = 1$	6.36	1000	63.6	100	636	400	3/t	250	1446	1000

pronounced compared to other explicit methods such as gradient descent.

We now give further considerations to variable damping. In Figure 3.6 we show the initial condition and final converged result for five separate damping experiments. Note that the final result remains the same regardless of the damping coefficient that is chosen for a. From the linear analysis in [34], and our choice of $\lambda = 1000$ and $\beta = 1$ for the two tuning parameters we have $a = \sqrt{\lambda + \beta\pi^2}$ giving an optimal damping of 63.6. In Figure 3.5 we compare an optimally damped system to a below optimal, above optimal, Nesterov, and critically damped system. The damping coefficients, and convergence times in iterations are given in Table 3.2. The below optimal and above optimal are each one order of magnitude away from the optimal damping, respectively, and the critical damping is the point at which the 2nd order accelerated scheme is equivalent to gradient descent, i.e the point at which the damping completely cancels



Initial Lena (a) Converged Result (b)

Figure 3.6: Initial (a) and Final (b) condition of denoising experiment with variable damping. The final result does not change between fixed damping and Nesterov only the number of iterations required.

out the momentum leaving only a first order descent for the PDE. While the optimal damping will always give the fastest convergence for the PDE, if one is uncertain of the optimal damping then using a greater than optimally damped but less than critically damped system will yield reasonable performance. While the Nesterov damping does converge faster than the above optimally damped example in Figure 3.5, the increasing damping as a function of time will yield degraded performance and would likely necessitate an additional stopping criteria. Although the below optimally damped system is initially faster than the optimally damped system, it is subject to large oscillations in energy, which while they do converge greatly slows down the final convergence time.

3.3.3 Image Deconvolution

One of the next experiments that we undertook with this new framework was the application of PDE acceleration to regularized deconvolution. Starting from the standard Beltrami-norm energy functional used for non-blind kernel restoration we write:

$$E(u) = \int_\Omega \frac{\lambda}{2}(K * u - g)^2 + \frac{1}{\beta}\sqrt{1 + ||\beta\nabla u||^2}\ d\mathbf{x} \tag{3.82}$$

where u is the current iteration of the image, K is the linear operator, in this case the known blurring function, g is the initial blurry image, β is the Beltrami regularizer and λ is the fidelity term. From this we can write the energy gradient as:

$$\nabla E = \lambda K^T * (K * u - g) - \nabla(\frac{\beta\nabla u}{\sqrt{1 + ||\beta\nabla u||^2}}) \tag{3.83}$$

In this case the nonlinear variational gradient decomposes as in (Equation 3.55) to the form[6]

$$\begin{aligned} \nabla E = \lambda K^T * (K * u - g) - \underbrace{\frac{\beta}{\sqrt{1 + \|\beta\nabla u\|^2}}}_{c}(\nabla \cdot \nabla u - u_{\eta\eta}) \\ - \underbrace{\frac{\beta}{\left(\sqrt{1 + \|\beta\nabla u\|^2}\right)^3}}_{d}\ u_{\eta\eta} \end{aligned}$$

The accelerated PDE (technically an integral partial differential equation with the convolution) takes the quasilinear form

$$u_{tt} - \nabla \cdot \left(\frac{\beta\nabla u}{\sqrt{1 + \|\beta\nabla u\|^2}}\right) + au_t = \lambda K^T * (g - K * u) \tag{3.84}$$

Note that both coefficients c and d are bounded by β (an upper bound which is

[6] Recall $\eta \doteq \frac{\nabla u}{\|\nabla u\|}$ denotes the unit vector along the gradient direction of u.

actually reached in both cases at any point and time where $\nabla u(x,t) = 0$), and that $\max|\mathrm{DFT}(K)| = 1$ by our assumption that K attenuates while preserving the mean. Plugging this into (Equation 3.58) yields

$$z_{\max} \leq \lambda + \frac{4N\beta}{\Delta x^2} \tag{3.85}$$

The discretization of the Laplacian $\nabla(\frac{\beta\nabla u^n}{\sqrt{1+||\beta\nabla u^n||^2}})$ is given as follows, consider a consistent discretization which converges as $\nabla u \to 0$ to the central differences approximation of the $\beta - scaled$ Laplacian $\beta\nabla\cdot\nabla u \approx \beta\sum_{k=1}^{N}\frac{u^n_{\alpha+e_k}-2u^n_\alpha+u^n_{\alpha-e_k}}{\Delta x^2}$ with spatial step size Δx in each direction. The multi-index subscript notation is given as follows:

$$\alpha = Z^N e_k = (\delta_{1k}, \delta_{2k}, ..., \delta_{Nk}) \tag{3.86}$$

where Z^N indicate each of the grid locations and where e_k is the additive multi-index. Here δ_{jk} is used as the standard Kronecker delta function. We let $D^2_{\beta,\Delta x}u^n$ denote the discretization of $\nabla\cdot\left(\frac{\beta\nabla u}{\sqrt{1+\|\beta\nabla u\|^2}}\right)$. Substituting into Equation 3.39 yields the 2nd-order explicit update:

$$\Delta u^n = \frac{2-a\Delta t}{2+a\Delta t}\cdot\Delta u^{n-1} - \frac{2\Delta t^2}{2+a\Delta t}\left(\lambda K^T*(K*u^n-g) - \nabla(\frac{\beta\nabla u^n}{\sqrt{1+||\beta\nabla u^n||^2}})\right) \tag{3.87}$$

Similar update schemes can be written for any of the desired discretization methods see Appendix E, Equation E.1, Equation E.2, Equation E.3, and Equation E.4. An interesting aside is that if we consider the limit as $\beta \to \infty$, the Beltrami regularization penalty converges to the total variation penalty.

$$E(u) = \int_\Omega \frac{\lambda}{2}(K*u-g)^2 + \|\nabla u\|\,dx \tag{3.88}$$

with a non-linear variational gradient (Equation 3.55) that decomposes as

$$\begin{aligned}\nabla E &= \lambda K^T * (K * u - g) - \nabla \cdot \left(\frac{\nabla u}{\|\nabla u\|} \right) \\ &= \lambda K^T * (K * u - g) - \underbrace{\frac{1}{\|\nabla u\|}}_{c} \left(\nabla \cdot \nabla u - u_{\eta\eta} \right).\end{aligned}$$

The accelerated PDE now takes the form of the nonlinear wave equation

$$u_{tt} - \nabla \cdot \left(\frac{\nabla u}{\|\nabla u\|} \right) + a u_t = \lambda K^T * (g - K * u). \tag{3.89}$$

In this case, the coefficient d vanishes, but the coefficient c no longer has a finite upper bound. Plugging this into (Equation 3.58) yields an infinite upper bound for the maximum gradient amplifier if at any point and time $\nabla u(x, t) = 0$. Otherwise, by our earlier assumption on K (see subsection 3.3.1) we obtain

$$\lambda \leq z_{\max} \leq \lambda + \frac{4(N-1)}{\Delta x^2 \min \|\nabla u\|}. \tag{3.90}$$

For the explicit second order accelerated scheme, this ensures the sufficient condition $\Delta t \leq \frac{2}{\sqrt{\lambda + \frac{4(N-1)}{\Delta x^2 \min \|\nabla u\|}}}$ for a stable step. If we fix Δt, we may rearrange this inequality to obtain an equivalent sufficient condition

$$\min \|\nabla u\| \geq \frac{N-1}{\Delta x^2} \frac{4 \Delta t^2}{4 - \lambda \Delta t^2}$$

which takes the form of a lower bound on the spatial gradient. Here an interesting nonlinear dynamic occurs to keep the implementation stable by preventing initiated instabilities from growing unbounded. If the spatial gradient falls below this lower bound and instabilities begins to propagate at one or more frequencies, they will eventually cause the spatial gradient to rise above the guaranteed stable lower bound at which point the instabilities will cease growing. In the absence of a kernel K, the fastest growing instability will occur at the highest digital frequency in each grid direction $\omega = (\pi, \ldots, \pi)$ which corresponds to oscillations between immediately adjacent grid-points, this in turn will most rapidly increase the discrete difference approximations of $\|\nabla u\|$. In the presence of a strongly smoothing kernel, the fastest growing instability may occur at lower digital frequencies, thereby causing a low-grade ringing effect, with several grid-points per period, until the amplitude of the oscillation is large enough to drive adjacent pixel differences back over the lower bound for $\|\nabla u\|$. A similar phenomenon occurs with both the first-order and semi-implicit schemes (and even with gradient descent), making all these schemes stable independently of the regularizer coefficient c. As such, purely for stability considerations alone, the necessary step size constraint will be connected to the lower bound λ of the gradient amplifier $z_{\max}$ rather than its upper bound in (Equation 3.90). This yields the following necessary conditions for stability.

$$\text{Gradient Descent: } \Delta t \leq \frac{2}{\lambda} \tag{3.91}$$

$$\text{1-Order Accelerated: } \Delta t \leq \sqrt{\frac{4}{\lambda} + \left(\frac{a}{\lambda}\right)^2} + \frac{a}{\lambda} \tag{3.92}$$

$$\text{2-Order Accelerated: } \Delta t \leq \frac{2}{\sqrt{\lambda}} \tag{3.93}$$

$$\text{Semi-Implicit: } \Delta t \leq \frac{2}{\sqrt{3\lambda}} \tag{3.94}$$

However, the schemes may only converge under these constraints in an oscillatory sense with a fluctuating level of "background noise" whose amplitude will depend upon the value of Δt.

We may exploit the behavior of this non-linear stabilizing effect to obtain a more useful time step constraint by plugging in a minimal acceptable value of $\|\nabla u\|$ for the final reconstruction into the stability condition for Δt. A natural way to approach this is by exploiting a quantization interval Q for the digital representation of u together with the following discrete approximation bounds for $||\nabla u\|$.

$$
\begin{aligned}
\min \|\nabla u\| &= \min_{\alpha} \sqrt{\sum_{k=1}^{N} \left(\frac{u_{\alpha+e_k} - u_{\alpha}}{\Delta x}\right)^2} \\
&\geq \sqrt{N \min_{\alpha,k} \left(\frac{u_{\alpha+e_k} - u_{\alpha}}{\Delta x}\right)^2} \\
&= \frac{\sqrt{N}}{\Delta x} \min_{\alpha,k} |u_{\alpha+e_k} - u_{\alpha}|
\end{aligned}
$$

If we now determine that instability related distortions confined to a single quantization interval Q between neighboring pixels are acceptable, we substitute

$$
\min \|\nabla u\| \to \frac{\sqrt{N}}{\Delta x} Q
$$

into the upper bound for (Equation 3.90) to obtain

$$
z_{\max} \leq \lambda + \frac{4(N-1)}{Q\Delta x\sqrt{N}} < \lambda + \frac{4\sqrt{N}}{Q\Delta x} \tag{3.95}
$$

within the desired stable regime for $\|\nabla u\|$. This in turn gives rise to the schemes in Appendix F (Equation F.1)–(Equation F.4), where $D^2_{\Delta x}u^n$ denotes the discretization of $\nabla \cdot \left(\frac{\nabla u}{\|\nabla u\|}\right)$. Now we present experimental results where in order to evaluate our accelerated second-order descent scheme we compared against two other state of the art deconvolution algorithms Chambolle-Pock Primal Dual and and L1 ADMM [7, 9]; see Figure 3.7. These results are improved from what was reported in [14] due to a change that made the input image fully symmetric and reduced ringing from the Fast Fourier Transform (FFT). In this experiment, see Figure 3.7, we are quite competitive in the quality of the restored image out performing both Primal Dual and L1 ADMM.

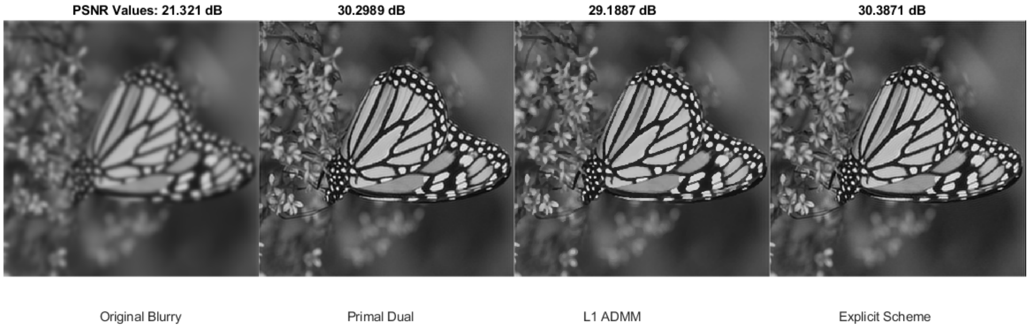


Figure 3.7: Deblurring of a butterfly image using the explicit accelerated PDE scheme. Performance is compared to two state of the art methods (final signal-to-noise ratios are shown for restoration.)

3.3.4 Applications for Compressed Sensing

Another application of PDE acceleration is in the field of compressed sensing. A common problem in the medical domain is the reduction of radiation dosage for CT scanners. One of the approaches, for using less radiation, is to reduce the number of x-ray projections that are acquired through the scanner's sensors [125]. This insufficient data collection results in a highly under sampled image with visible streaking. To recover the undersampled image, researchers often employ one of two edge-preserving TV based reconstruction algorithms. Typical approaches to the minimization of this TV problem are often conjugate gradient or gradient descent [126, 72].

If we treat the Fourier back-projection as a replacement for our linear convolution operator then the cost functional for this problem can be formulated almost identically to the case of the regularized deconvolution problem. For our work we again consider equation Equation 3.82, restated here for convenience.

$$E(u) = \int_{\Omega} \frac{\lambda}{2}(K * u - g)^2 + \frac{1}{\beta}\sqrt{1 + ||\beta \nabla u||^2} \;\; d\mathbf{x}$$

This problem however is slightly harder than the problem of deconvolving an image with a known blurring kernel. In the case of a general Gaussian kernel we have the benefit that $K * u \neq g$ for $u = g$. However in the case of Fourier back-projection this is not true. Treating the linear operator K as our Fourier back-projection will lead to $K * u = g$ for $u = g$. Although our case is not hopeless as we are convolving on the evolving entity u^n; this problem is not nearly as well suited for gradient descent. Principally without the presence of a strong regularizer the gradient of the energy will be small and little to no improvement in the resulting image quality will be made. Thus we will have to employ a strong regularizing term to drive the evolution of the PDE. Example images to demonstrate the reconstruction problem for Fourier back-projection are provided in Figure 3.8. Note that the images are progressively degraded with a drop in the number of projections.

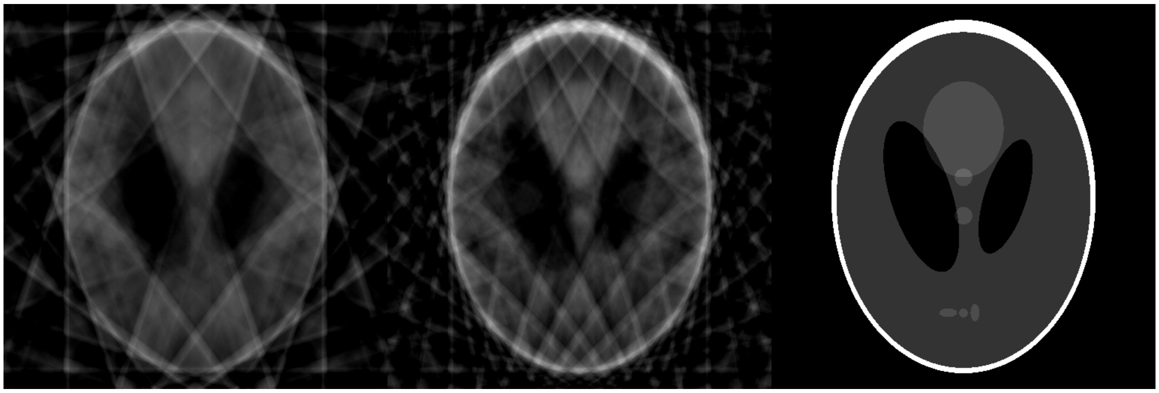


Figure 3.8: Reconstruction of a CT image using varying numbers of projections for the Fourier back projection algorithm. Number of projections used from left to right: 10, 20 with the original image given on the right hand side.

With the caveat of the descent scheme in mind, we write an almost identical update equation to Equation 3.87

$$\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t} \cdot \Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t}\left(\lambda K^T * (K * u^n - g) - \nabla\left(\frac{\beta \nabla u^n}{\sqrt{1 + ||\beta \nabla u^n||^2}}\right)\right)$$

with the primary difference being that K and K^T correspond to the Fourier back-projection algorithm. From here we show the attempted recovery of the CT slice on the preceding example see Figure 3.8.

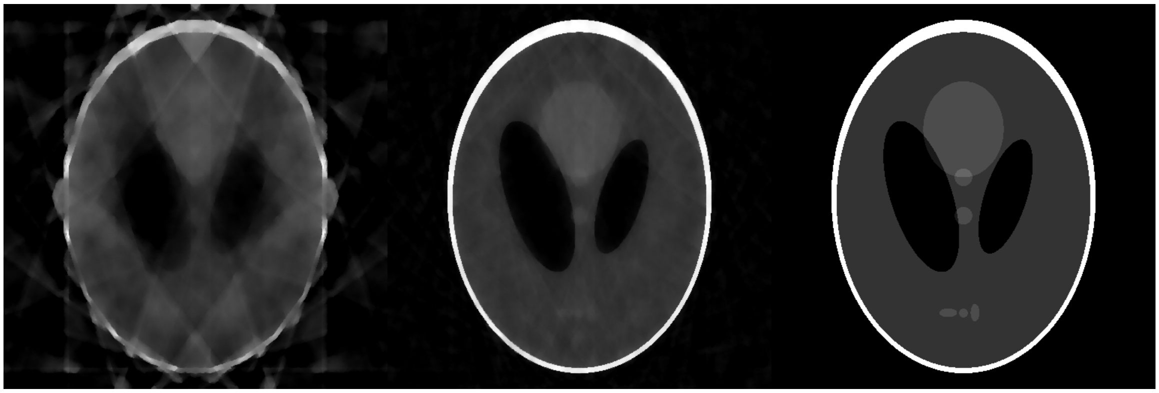


Figure 3.9: Recovered CT images (using PDE acceleration) from Figure 3.8. The original image is given on the right hand side for comparison. The parameters for this experiment were $\lambda = 10^6, \beta = 100$ and the number of iterations were $T = 2500$.

Note that the recovery is excellent and significantly more successful in the case of increasing numbers of projections. Additionally the strong regularizing penalty $\beta = 100$ is necessary to drive the PDEs. We also note for large values of β the Beltrami regularizer approximates an L_1 TV norm. The image improvements for the experiment in Figure 3.9 are given in Table 3.3.

Table 3.3: Performance of Second-Order PDE Acceleration scheme for recovering undersampled CT images. PSNR was computed by comparing to the original clean image.

Projections	**Initial PSNR (dB)**	**Final PSNR (dB)**
10	16.00	18.04
20	17.51	32.15

We compared the experiment with PDE acceleration to the same reconstruction using the first-order primal dual method. The two methods are quite close in performance with primal dual slightly out-performing PDE acceleration. One minor benefit of PDE acceleration was that we could significantly increase the penalty on the data term while maintaining a reasonable step size. In the first order primal-dual method the step size is proportional to $\frac{1}{\lambda}$ which limits the fidelity coefficient.

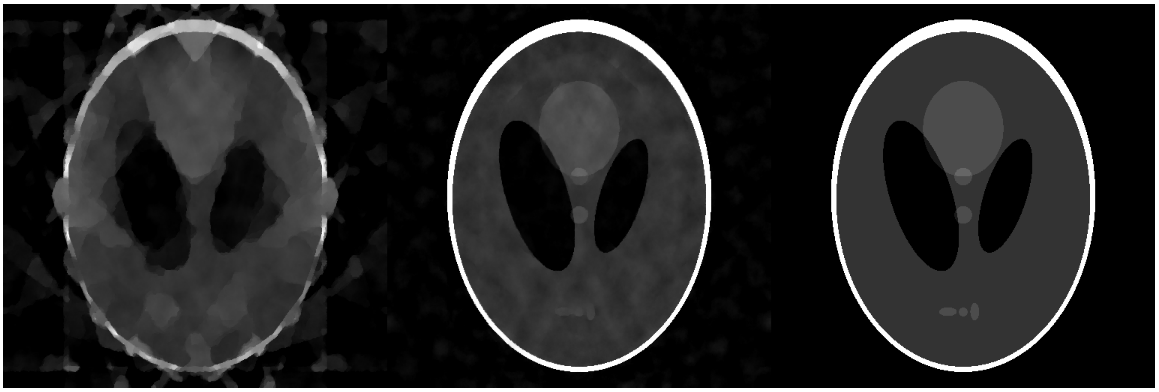


Figure 3.10: Recovered CT images (using first-order primal-dual) from Figure 3.8. The original image is given on the right hand side for comparison. The parameters for this experiment were $\lambda = 10^3, \beta = 100$ and the number of iterations were $T = 2500$.

Table 3.4: Performance of first order Primal Dual scheme for recovering undersampled CT images. PSNR was computed by comparing to the original clean image.

Projections	**Initial PSNR (dB)**	**Final PSNR (dB)**
10	16.00	17.78
20	17.51	31.40

Although this work is limited, we argue that it provides a compelling case for additional experimentation with PDE acceleration for compressed sensing applications.

3.3.5 Poisson's Equation

Another application for PDE acceleration is as an accelerated solver for the Poisson equation. Here we denote the Laplace operator as ∇^2 and write the functional for the Poisson equation as:

$$\int_\Omega (\frac{1}{2}(|\nabla(u)|^2) + \frac{\lambda}{2}(u(x) - f(x))^2)dx \tag{3.96}$$

One option for this solving problem is to just consider the Laplacian and the non-homogenous term:

$$\nabla^2 u = f \tag{3.97}$$

This is slightly different from what would be obtained by taking the first variation. In the two dimensional case we can write the finite difference approximation for Poisson's equation (Equation 3.97) as:

$$\frac{u_{j+1,k} + u_{j-1,k} + u_{j,k-1} + u_{j,k-1} - 4u_{j,k}}{(\Delta x)^2} = \rho_{j,k} \tag{3.98}$$

This can be written as the solution to a matrix inversion problem:

$$A \cdot \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ \vdots \\ u_{N,1} \\ u_{1,2} \\ \vdots \\ u_{N,2} \\ \vdots \\ u_{N,N} \end{bmatrix} = \Delta x^2 \begin{bmatrix} \rho_{1,1} \\ \rho_{2,1} \\ \vdots \\ \rho_{N,1} \\ \rho_{1,2} \\ \vdots \\ \rho_{N,2} \\ \vdots \\ \rho_{N,N} \end{bmatrix} \tag{3.99}$$

where

$$A = \left[\begin{array}{ccccccccccccccc} \ddots & & & & \ddots & \ddots & \ddots & & & & \ddots & & & & \\ 0 & 1 & 0 & \dots & 0 & 1 & -4 & 1 & 0 & \dots & 0 & 1 & 0 & \dots & \\ \dots & 0 & 1 & 0 & \dots & 0 & 1 & -4 & 1 & 0 & \dots & 0 & 1 & 0 & \dots \\ & \dots & 0 & 1 & 0 & \dots & 0 & 1 & -4 & 1 & 0 & \dots & 0 & 1 & 0 \\ & & & & \ddots & & & \ddots & \ddots & \ddots & & & & & \ddots \end{array}\right] \tag{3.100}$$

This calculation can be expressed as an $Ax = b$ problem where A and b are known. In this problem A is singular and so x is typically solved for using some type of descent scheme such as the Jacobi method or a relaxation algorithm such as Successive over-relaxation (SOR) or Gauss-Seidel (GD). To write the second order accelerated descent scheme we employ the same methodology as was used in the preceding sections. First we consider the gradient of the energy:

$$\Delta E = \lambda(f - u) - \nabla^2 u \tag{3.101}$$

and then we plug the energy of the gradient into Equation 3.39. This yields our second order update equation:

$$\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t} \cdot \Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t}\Big(\lambda(f - u) - \nabla^2 u\Big) \tag{3.102}$$

We now apply this to an example Poisson equation and compare against several commonly used solvers. Consider the two-dimensional Poisson problem:

$$\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} = 1.25 \cdot exp(x + \frac{y}{2}) \tag{3.103}$$

where $x, y \in [0, 1]$. For the grid points we consider $m = 20$ linearly spaced points along each dimension. We write the matrix $f_{i,j} = 1.25 \cdot exp(x_{i,j} + \frac{y_{i,j}}{2})$ where i, j are the respective indices in each dimension. We seek to solve the problem:

$$Au = f \tag{3.104}$$

where $A \in \mathbb{R}^{(m^2 \times m^2)}$ is the singular sparse matrix corresponding to the 5-point discretization of the Laplacian.

The performance results of our optimization methods are given in Table 3.5 and Table 3.6. In the first experiment we considered sparse matrices of size 400 x 400 elements. In Table 3.5 we can clearly observe that PDE acceleration outperformed both the Jacobi method and GD. Although the SOR method did use less iterations it required a more complex calculation and as such had nearly double the run time

Table 3.5: Performance of Second-Order PDE Acceleration scheme for solving Poisson equation. Our method is compared against standard solvers of which the Jacobi Method is the most comparable to gradient descent.

Method	**Iterations**	**Run Time (ms)**
PDE Acceleration	92	0.51
Jacobi Method	998	5.0
SOR	58	1.1
Gauss-Seidel	490	6.5

of the PDE accelerated method. We now consider the case of a large sparse matrix $m = 100$, which yields a 10000 x 10000 element sparse matrix.

Table 3.6: Performance of Second-Order PDE Acceleration scheme for solving Poisson equation. Notice that the PDE method out competes state of the art optimizers even without the benefit of GPU parallelization.

Method	**Iterations**	**Run Time (sec)**
PDE Acceleration	577	0.042
Jacobi Method	26398	2.765
SOR	366	0.125
Gauss-Seidel	12947	4.093

In Table 3.6 we can again clearly see that the performance results from the earlier experiment holds. PDE acceleration in fact is nearly three times faster than the SOR method. Also note that traditional SOR cannot be parallelized without alternating the grid update. This restriction is not true of PDE acceleration and offers another potential avenue of speed up in highly parallelized applications such as GPU processing.

3.4 Optical Flow

3.4.1 General Implementation

The next application of the PDE acceleration framework is to the problem of optical flow. In the optical flow problem the general idea is that we want to determine how pixels moved from one image to the next. We write the energy function motivated from Lucas and Kanade's approach [66] as

$$U(\phi) = \frac{1}{2} \int_{\mathbb{R}^n} |I_1(\phi(x)) - I_0(x)|^2 \, dx + \frac{1}{2} \alpha \int_{\mathbb{R}^n} |\nabla(\phi(x) - x)|^2 \, dx \tag{3.105}$$

where ϕ is the forward map from image I_1 to I_0 and α is the regularizing term. The first term is the data fidelity which measures how close ϕ deforms I_1 back to I_0 through the squared norm, and the second term penalizes non-smoothness of the displacement field, given by $\phi(x) - x$ at the point x. Notice that the potential is a function of only the mapping, ϕ and not the velocity.

We now compute the functional gradient of U with respect to the mapping ϕ, denoted by the expression $\nabla U(\phi)$. This gradient is defined by the relation (see Appendix G) $\delta U \cdot \delta\phi = \int_{\phi(\mathbb{R}^n)} \nabla U(\phi) \cdot \delta\phi \, dx$, i.e., the functional gradient satisfies the relation that the $\mathbb{L}^2$ inner product of it with any perturbation $\delta\phi$ of ϕ is equal to the variation of the potential U with respect to the perturbation $\delta\phi$. With this definition, one can show that (see Appendix G):

$$\nabla U(\phi) = [(I_1 - I_0 \circ \psi)\nabla I_1 - \alpha(\Delta\phi) \circ \psi] \det \nabla\psi, \tag{3.106}$$

where det denotes the determinant. We can also see that the gradient defined on the un-warped domain is

$$\widetilde{\nabla} U(\phi) = (I_1 \circ \phi - I_0)\nabla I_1 \circ \phi - \alpha\Delta\phi, \tag{3.107}$$

We write the equivalent but mathematically more convenient energy of the gradient as:

$$-\nabla U = -\nabla I \circ \phi \cdot (I_1(\phi) - I_0(x)) + \alpha\nabla^2\phi(x) \tag{3.108}$$

where $\nabla I \circ \phi = \left[\begin{smallmatrix} \frac{\delta I(\phi)}{\delta x} \\ \frac{\delta I(\phi)}{\delta y} \end{smallmatrix} \right]$ denotes the spatial gradient of the projected image $I_1(\phi)$. By substituting into Equation 3.39 and changing u to ϕ this allows us to write the new descent equation for computing the optical flow:

$$\Delta \phi^n = \frac{2 - a\Delta t}{2 + a\Delta t} \cdot \Delta \phi^{n-1} + \frac{2\Delta t^2}{2 + a\Delta t} \cdot \Big(-\nabla I_x(\phi(x)) \cdot (I_1(\phi) - I_0(x)) + \alpha \nabla^2 \phi(x) \Big) \quad (3.109)$$

The maximum stable time step Δt for the scheme Equation 3.109 can be derived using Von Neumann analysis. From [14], Δt should be chosen as $\Delta t < \frac{2}{\sqrt{Z_{max}}}$, where Z_{max} is the maximum value over all frequencies of the Fourier transform of the linearization of the homogeneous part of the gradient, ∇U. In the case of the Horn & Schunck energy, this corresponds to

$$\Delta t < \frac{2}{\sqrt{1 + 8\alpha}}, \quad (3.110)$$

where the above is an approximation and we assume that the image is normalized to 1.

Next we compute the optimal damping coefficient, a. To do this, we use results from [34], which computes the convergence rate of accelerated PDE as a function of the damping in the case that the energy is convex. The Horn & Schunck energy is not convex, however, the linearization of the gradient in the accelerated PDE corresponds to a convex energy that was analyzed in [34].

The original accelerated PDE is:

$$\partial_{tt}\phi + a\partial_t \phi - \alpha \Delta \phi + (I_1 \circ \phi - I_0) \nabla I_1 \circ \phi = 0. \quad (3.111)$$

We can linearize the non-linear term and compute the optimal damping. For simplicity (as we did not find much difference in the speed of overall convergence in our experiments), we simply treat the non-linear term as zero (which is true if ϕ is near the solution as $I_1 \circ \phi - I_0$ is close to zero; in practice we use a pyramid method where the solution is close to the optimal since it is initialized with the solution from the previous scale). In this case, the PDE reduces to

$$\partial_{tt}\phi + a\partial_t\phi - \alpha\Delta\phi = 0, \tag{3.112}$$

which is a vector-valued version of an equation analyzed in [34]. The optimal damping is given as

$$a = 2\sqrt{\alpha\mu_1} \tag{3.113}$$

where μ_1 is the first Neumann eigenvalue of the Laplacian. The eigenvalue can be approximated as $\mu_1 \approx \frac{\pi^2}{A}$ where A is the area of the image domain (width times height of the image). This gives the damping condition that we used for our experiments as:

$$a = 2\sqrt{\alpha\mu_1} \approx 2\sqrt{\frac{\pi^2\alpha}{A}}. \tag{3.114}$$

For our illustrated experiments we compare gradient descent against acceleration, damped and non-damped acceleration and conjugate gradient, one of the most popular optical flow methods [67] on the frequently evaluated middlebury data set [12]. In our results we find that we arrive at a similar local minimum but with significantly less computational time. Our method is nearly an order of magnitude faster than conjugate gradient. Both methods used the pyramid scheme with a resolution increase of a factor of 2 at each step in the pyramid.

3.4.2 Acceleration vs Gradient Descent

In our first set of optical flow experiments, we compare the discrete implementation of the Eulerian approach to accelerated optimization on the manifold of diffeomorphisms to standard (Riemannian L^2) gradient descent. This illustrates the potential performance improvements one can achieve by incorporating acceleration. When compared to gradient descent, acceleration requires only to update the velocity by the velocity and density evolution. These evolutions only require local updates.

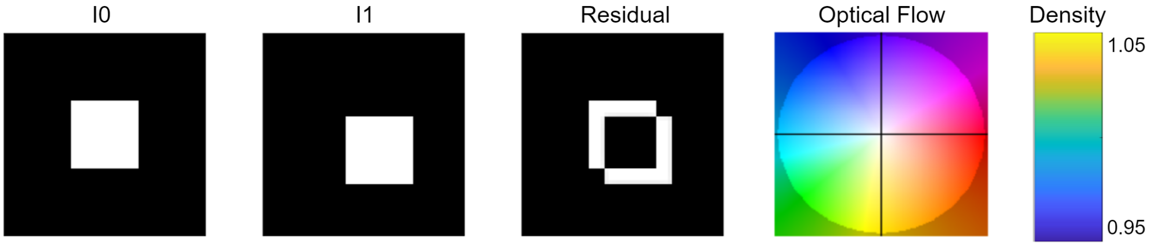


Figure 3.11: Illustrative experiment: The experiment (whose results are in Figure 3.13 and Figure 3.12) computes the optical flow (registration) between I_0 and I_1 using a common optical flow loss function. The initial residual ($|I_1 - I_0|$) is shown. The fourth image from left is a color code for the velocity (the direction of the velocity is indicated by the color). The fifth image is a color code for the mass density graphs used in Figure 3.13.

Consider the optimization problem presented in Figure 3.11. We show the evolution, which aims to register a square to a translated square (see Figure 3.12). We compare the evolutions for acceleration with and without damping, both which introduce oscillations, but the former dies down. The evolutions eventually determine a translation, even though the velocity can vary with pixel location. Notice the mass density evolves through non-uniform densities at times, indicating a non-trivial mass evolution impacting the dynamics. A comparison to gradient descent is shown in Figure 3.12, in particular showing that acceleration drastically speeds up convergence.

3.4.3 Acceleration vs Conjugate Gradient

A visual comparison illustrating the converged results for the two methods is provided in Figure 3.14 with a performance benchmark table provided in Table 3.7. Initially the performance gap between conjugate gradient and the accelerated gradient is small but widens significantly as the image resolution increases. By the final level of the pyramid PDE acceleration is almost an order of magnitude faster.

Table 3.7: Performance comparison of Linearized Optical Flow against Accelerated Optical Flow for each level of the pyramid. The performance improvement of accelerated optical flow is close to an order of magnitude. Both methods arrive at nearly the same local minima. AAE and AEE are average angular error and end point error respectively. Note the quantities above represent average values over all pairs of images in the dataset.

Middlebury Benchmark Results

Pyramid Level-1 (Res: 1/32)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	0.299	0.290	2.779	
Accelerated Optical Flow	**0.209**	0.290	2.774	1.430
Pyramid Level-2 (Res: 1/16)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	0.476	0.150	1.679	
Accelerated Optical Flow	**0.296**	0.150	1.685	1.608
Pyramid Level-3 (Res: 1/8)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	2.085	0.114	1.242	
Accelerated Optical Flow	**0.952**	0.115	1.249	2.191
Pyramid Level-4 (Res: 1/4)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	7.862	0.102	1.045	
Accelerated Optical Flow	**2.279**	0.102	1.046	3.450
Pyramid Level-5 (Res: 1/2)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	90.507	0.094	0.845	
Accelerated Optical Flow	**10.763**	0.094	0.851	8.409
Pyramid Level-6 (Res: 1)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	1131.373	0.090	0.701	
Accelerated Optical Flow	**114.307**	0.090	0.700	9.898
Cumulative (All Levels)	**Time to Converge (sec):**	**AAE (rad):**	**AEE (pixels):**	**Speed Up**
Linearized Optical Flow	1232.602	0.090	0.701	
Accelerated Optical Flow	**128.806**	0.090	0.700	9.569

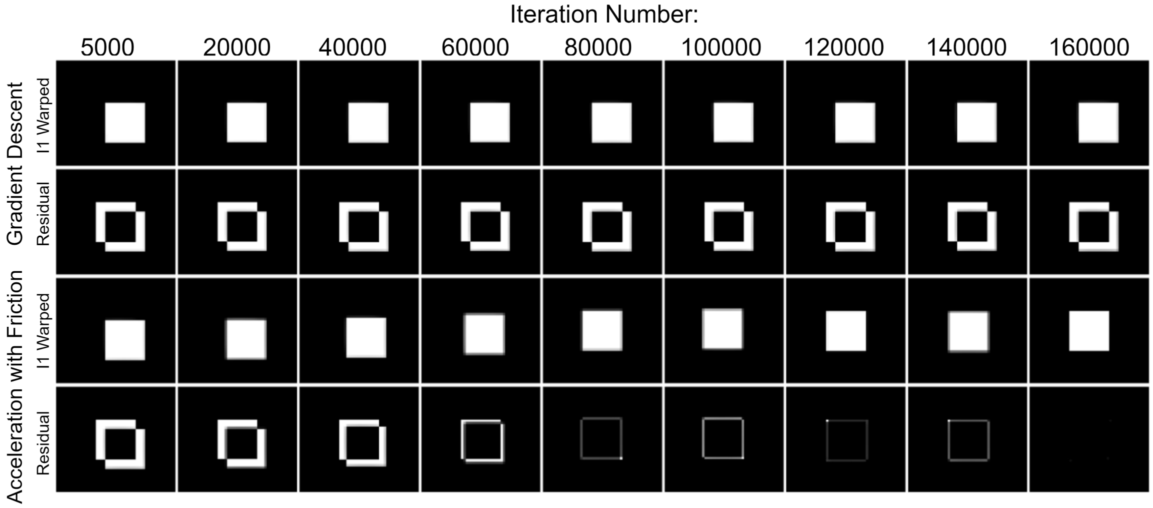


Figure 3.12: Comparison of evolutions of gradient descent and accelerated gradient descent. $I_1(\phi)$ and the residual are shown throughout the evolution. As can be seen, acceleration converges in far fewer iterations (gradient descent eventually converges, but much later in time).

The optical flow data set [12] that was used in these experiments is the Middlebury dataset, a standard benchmark for optical flow, and can be found on https://vision.middlebury.edu/flow/data/. Images ranged in resolution from 420 x 380 to 640 x 480, and consist of 7 real scenes with camera motion as well as object motion and deformation. The ground truth dense optical flow is also provided in this data set. The accuracy of the optical flow on this data set is measured with the average angular error (AAE), which measures the average angular difference between the result and ground truth displacement vectors, and the average end point error (AEE), which measures the average difference between end points of the displacement vectors of the result and ground truth. The regularization was kept constant throughout the entire experiment for every image. We fix the coefficient on the regularizer α at 0.04 for both accelerated and linearized optical flow, which leads to the optimal accuracy for both methods.

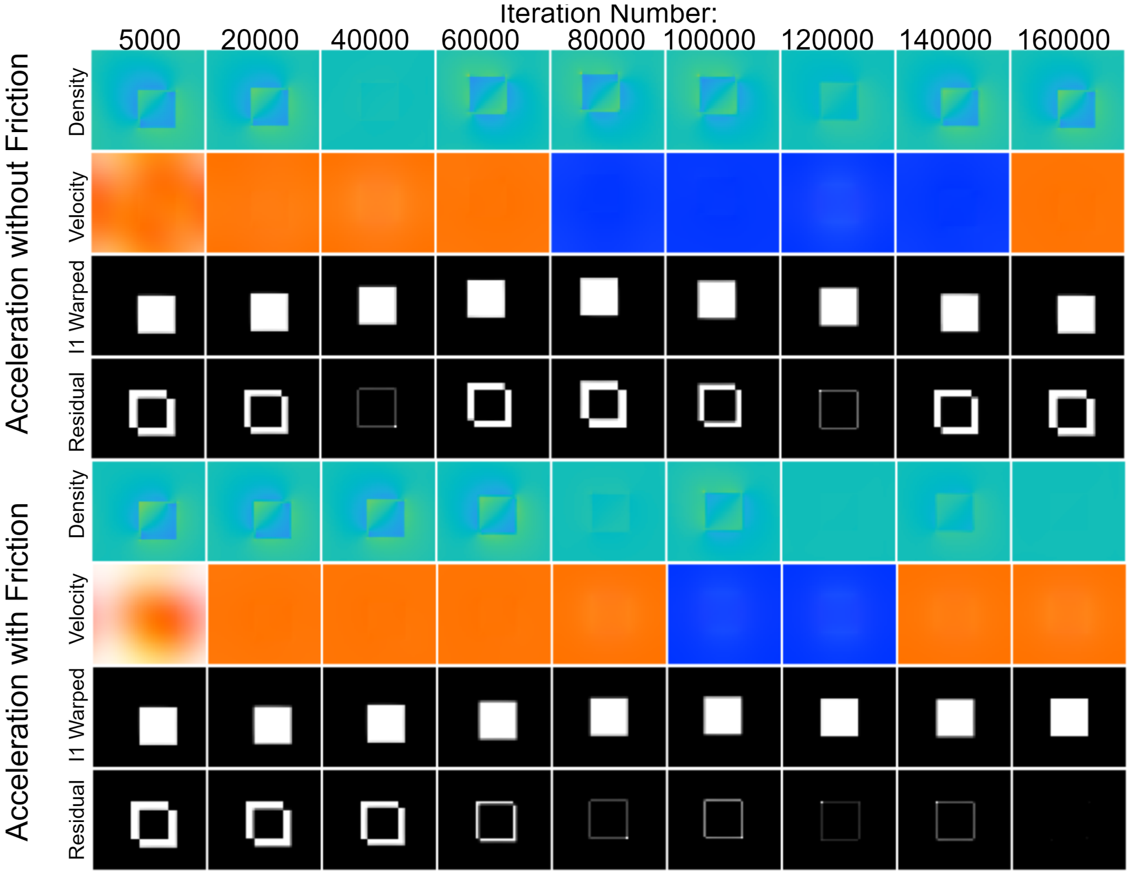


Figure 3.13: Comparison of evolutions of accelerated optimization with and without friction. The four rows are the density $\rho_t \# \phi_t$, velocity $v \circ \phi_t$, image warp $I_1 \circ \phi$ and residual $|I_0 - I_1(\phi)|$ for the undamped and damped accelerated descents over various iterations. Notice that the undamped descent overshoots the target and switches directions as evidenced by the shift in the velocity from orange to blue. The addition of a friction term kills the oscillations, allows convergence and for the minimization of the residual. Notice that in both cases, the mass moves within and around the square in non-trivial ways. Each are initialized with a constant density and at convergence, the density is also constant.

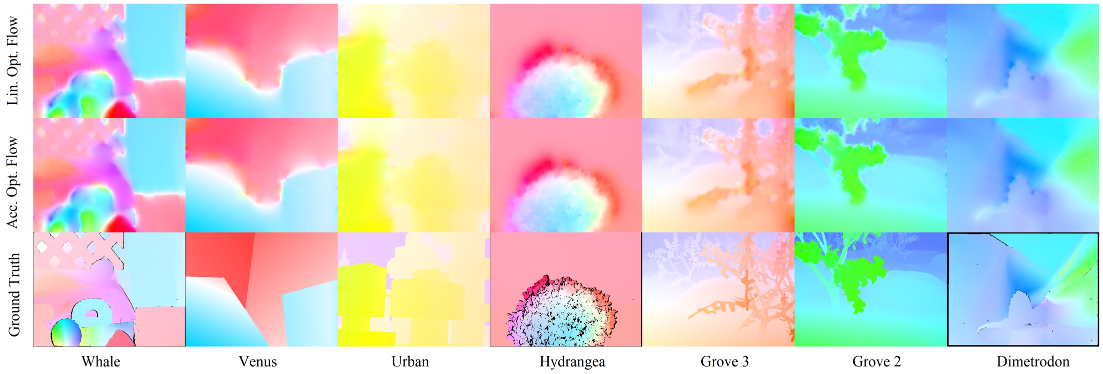


Figure 3.14: Converged results for Linear Optical Flow and Accelerated Optical Flow run on the Middlebury data set. Both methods converge to the same local minimum, with the advantage of accelerated being speed. Note the black areas indicate occlusion, which are excluded from error computation according to the benchmark. Seven image pairs were used for the experiment.

3.5 Active Contours

3.5.1 Formulation

We now illustrate the steps in the process for developing PDE based accelerated optimization schemes for the specific case of geometric active contours. The resulting coupled PDE evolutions will retain the parameterization independent property of gradient descent based active contours models and will therefore remain amenable to implicit implementation using Level Set Methods [127].

We begin, however, by reviewing some basic differential contour evolution properties that will be useful in deriving accelerated active contour formulations. In particular, it is useful to understand any contour evolution behavior in terms of its local geometric frame, consisting of the unit tangent and normal vectors.

Let $C(p,t)$ denote an evolving curve where t represents the evolution parameter and $p \in [0,1]$ denotes an independent parameter along each fixed curve. The unit tangent, unit normal, and curvature will be denoted by $T = \frac{\partial C}{\partial s}$, N, and κ respectively, with the sign convention for κ and the direction convention for N chosen to respect the planar Frenet equations $\frac{\partial T}{\partial s} = \kappa N$ and $\frac{\partial N}{\partial s} = -\kappa T$, where s denotes the time-dependent arclength parameter whose derivative with respect to p yields the parameterization speed $\frac{\partial s}{\partial p} = \left\| \frac{\partial C}{\partial p} \right\|$.

Letting α and β denote the tangential and normal speeds of the evolving curve[7],

$$\frac{\partial C}{\partial t} = \alpha T + \beta N \tag{3.115}$$

[7] Note that the instantaneous geometric deformation of the curve is determined exclusively by the normal speed β, and that gradient flows for geometric active contours can all be formulated such that the tangential speed α vanishes. We will see later that the same is possible for accelerated flow models as well.

the frame itself can be shown to evolve as follows.

$$\frac{\partial T}{\partial t} = \left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right) N, \qquad \frac{\partial N}{\partial t} = -\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right) T \tag{3.116}$$

Differentiating the velocity decomposition (Equation 3.115) with respect to t, followed by the frame evolution (Equation 3.116) substitution, yields the acceleration

$$\frac{\partial^2 C}{\partial t^2} = \left(\frac{\partial \alpha}{\partial t} - \beta\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right)\right) T + \left(\frac{\partial \beta}{\partial t} + \alpha\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right)\right) N \tag{3.117}$$

which may be rewritten as the following two scalar evolution equations for the tangential and normal speeds, in terms of the tangential and normal components of the contour acceleration, respectively.

$$\frac{\partial \alpha}{\partial t} = \frac{\partial^2 C}{\partial t^2} \cdot T + \beta\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right), \qquad \frac{\partial \beta}{\partial t} = \frac{\partial^2 C}{\partial t^2} \cdot N - \alpha\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right) \tag{3.118}$$

.

For geometric active contours, we start by defining the potential energy $\mathbf{U}$ to be an originally provided energy functional E which depends only upon the geometric shape of the contour C (not its parameterization). Under these assumptions the first variation of the potential energy will have the following form, just as in (Equation 2.12) presented earlier in Equation 2.12, where fN denotes the backward local gradient force at each contour point.

$$\delta\mathbf{U} = -\int_C f\left(\delta C \cdot N\right) ds$$

Constant density model

To formulate an accelerated evolution model, we define a kinetic energy, which requires a notion of mass coupled with velocity. The simplest starting model would be one of constant mass density ρ (per unit arclength along the contour) and an integral of the squared norm of the point-wise contour evolution velocity[8].

$$\mathbf{T} = \frac{1}{2}\rho \int_C \left(\frac{\partial C}{\partial t} \cdot \frac{\partial C}{\partial t} \right) ds \tag{3.119}$$

Plugging this into the generalized action integral (Equation 3.1) and computing the Euler-Lagrange equation leads to our first, and simplest, accelerated model.

$$\underbrace{\frac{\partial^2 C}{\partial t^2}}_{\text{acceleration}} = \frac{\lambda k^2 t^{k-2}}{\rho} \underbrace{fN}_{\text{–gradient}} - \underbrace{\left(\frac{\partial^2 C}{\partial s \partial t} \cdot \frac{\partial C}{\partial s} \right) \frac{\partial C}{\partial t} - \frac{\partial}{\partial s} \left(\frac{1}{2} \left\| \frac{\partial C}{\partial t} \right\|^2 \frac{\partial C}{\partial s} \right)}_{\text{wave propagation terms}} - \underbrace{\frac{k+1}{t} \frac{\partial C}{\partial t}}_{\text{friction}} \tag{3.120}$$

If we start with zero initial velocity we can decompose this nonlinear second-order PDE into the following coupled system of nonlinear first order PDE's

$$\frac{\partial C}{\partial t} = \beta N, \qquad \frac{\partial \beta}{\partial t} = \frac{\lambda k^2 t^{k-2}}{\rho} f + \frac{1}{2}\beta^2 \kappa - \frac{k+1}{t}\beta \tag{3.121}$$

Since the contour evolution remains purely geometric (only in the normal direction N) we may also write down an implicit level set version of the coupled PDE system as follows

$$\frac{\partial \psi}{\partial t} = \hat{\beta} \|\nabla \psi\|, \qquad \frac{\partial \hat{\beta}}{\partial t} = \frac{\lambda k^2 t^{(k-2)}}{\rho} \hat{f} + \nabla \cdot \left(\frac{1}{2} \hat{\beta}^2 \frac{\nabla \psi}{\|\nabla \psi\|} \right) - \frac{k+1}{t} \hat{\beta} \tag{3.122}$$

where $\hat{f}(x,t)$ and $\hat{\beta}(x,t)$ denote spatial extensions of f and β respectively.

[8] A similar kinetic energy model in the context of the classical action $\mathbf{T} - \mathbf{U}$, for example, was used to develop dynamic geodesic snake models for visual tracking in [128]

3.5.2 PDE Acceleration vs Gradient Descent

In this section we illustrate the performance improvements of reformulating an existing active contour model into the accelerated framework and demonstrate favorable performance improvements even when comparing against recent alternative global strategies such as Chambolle-Pock. As the scope of this paper is not to propose or invent a particular active contour model, but rather an accelerated framework that can apply to variational models, we keep the 2D test images simple, such that popular binary region based active contour models (such as Chan-Vese) are well suited to the segmentation tasks. We demonstrate that these models, however, when lacking sufficient regularity (in this case the arc length penalty), become prone to getting trapped within unwanted local minimizers when implemented as standard gradient descent active contours. We show that even strategies such as Chambolle-Pock, which seek to minimize the global energy, still become numerically trapped within local minimizers when used with matching regularity. And while these alternative global minimizers can perform admirably on special classes of binary region based active contours they are not extendable with the same generality as the PDE acceleration framework presented here. We will see in these illustrative examples that simply applying the contour acceleration is sufficient to fix the sensitivity to local minimizers, drastically speed up the convergence of the region based active contours and without the need to abandon the active contour framework in favor of less generalizable global convex optimization methods. We also provide example extensions of the acceleration framework for non-convex problems, particularly the case of variational 3D reconstruction, where we show not only a dramatic speed-up in terms of run-time but also a better converged result for both toy and real data.

In Figure 3.15 we see three different initial contour placements (top, middle, bottom) evolving from left-to-right via the gradient flow PDE (Equation 2.13). Each gets trapped within a different local minimizer due to noise, all of which lie very

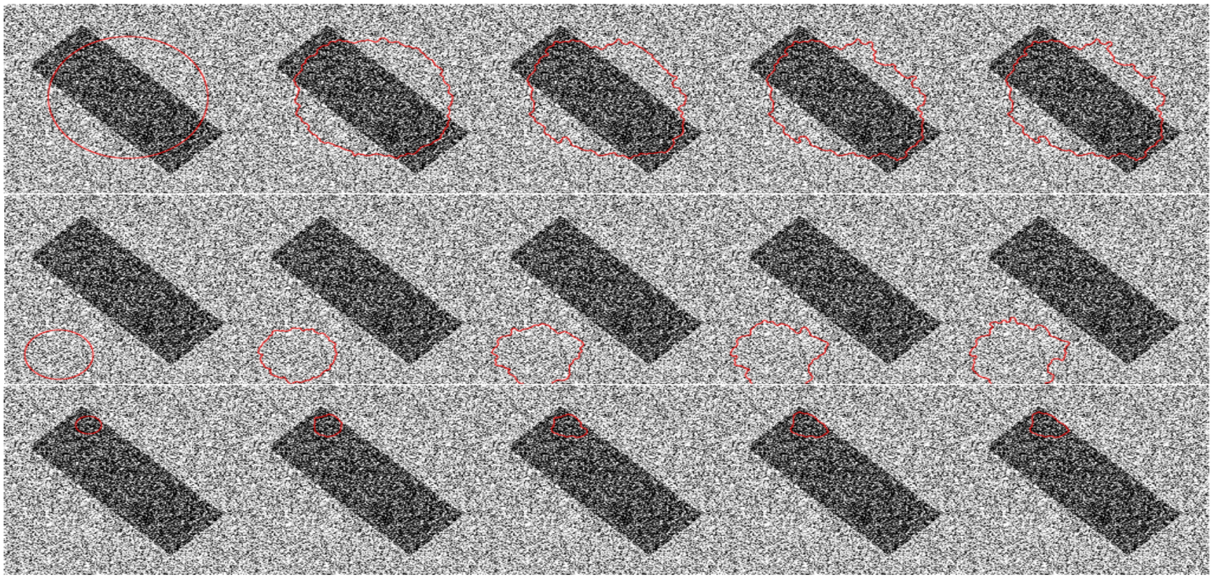


Figure 3.15: Different initial contours flowing into local energy minimizers

far away from the desired much deeper minimizer along the rectangular boundary. Of course, stronger regularizing terms could be added to the active contour energy functional to impose smoothness on the contour, thereby making it resistant to noise. However, the point of this experiment was to create an energy landscape littered with literally tens of thousands (perhaps even hundreds of thousands) of local minimizers in order to demonstrate the effects of acceleration. Furthermore, stronger regularization would sacrifice the ability to capture the sharp corners of the rectangle and increase the computational cost due to smaller resulting step size constraints in the PDE discretization.

We avoid both of these sacrifices by using the exact same active contour force f within the accelerated PDE system (Equation 3.121) instead. In Figure 3.16, we see the effect of applying accelerated contour evolution scheme with the same initial contour placements and same energy functional (no additional regularizing terms). In all three cases, the accelerated PDE system pushes the contour past the noise, driving it toward a more robust minimum along the rectangle edge.

In Figure 3.17 we see this same dramatic difference on a real seismographic image

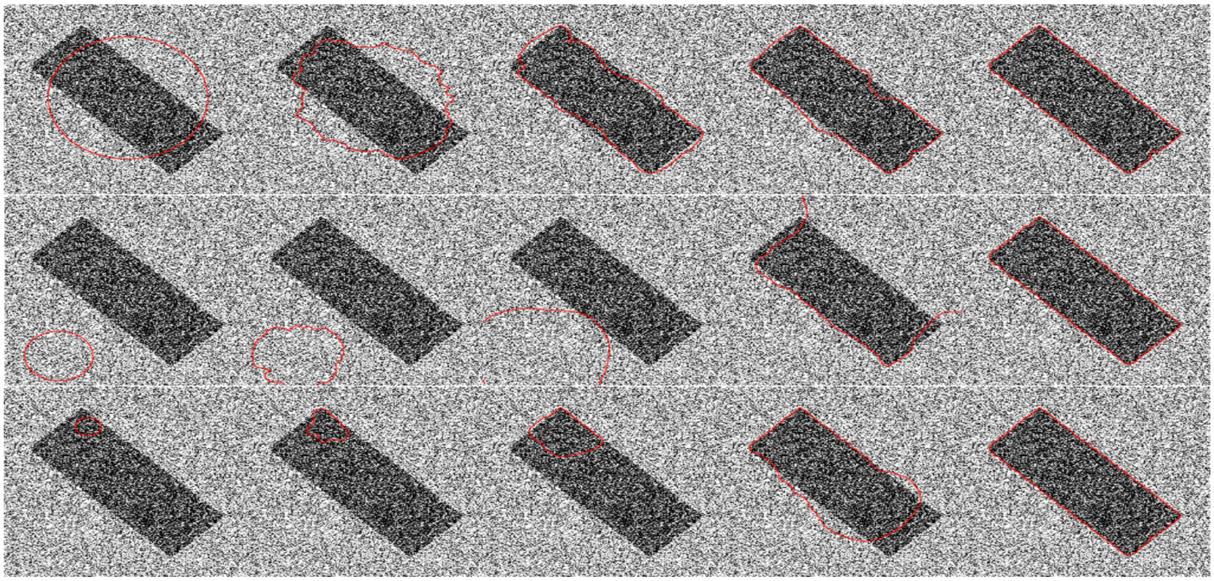


Figure 3.16: Accelerated active contours flowing past local minima

where we attempt to use an active contour to pull out the rather noisy ”core” of the recorded seismograph line. Along the left column we see four different initial contour placements, where the first three elliptical initializations, which are far from the desired segmented result, pose a considerable challenge to a classical gradient descent active contour. Minimal regularization is allowed here given the spikey nature of the signal, at least in cases where we wish to capture this fine scale level of detail.

In the middle column, we see the converged active contour results based on the standard gradient flow version of the evolution given by (Equation 2.13). Only in the last (bottom) case, is the segmented result reasonable.

In the last column, we see the converged result of the same active contour energy E and force f evolved using the accelerated PDE system (Equation 3.121). While there are very subtle differences in the final results (as can be see by the slight differences in the converged energy value), all four are nonetheless reasonable now even from the first three challenging initial contour placements.

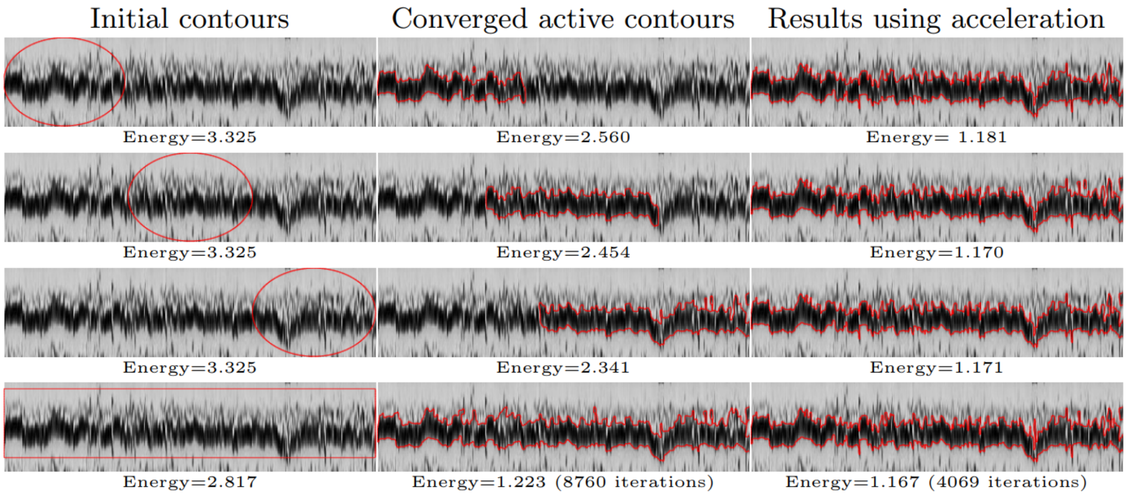


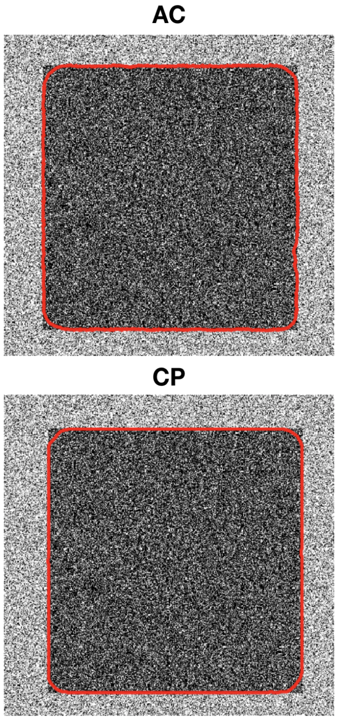
Figure 3.17: Non-accelerated (middle) vs. accelerated (right) active contour results for same four initializations (left) on a seismograph image. Cost functional values underneath.

3.5.3 PDE Acceleration vs Primal Dual

In Table 3.8, we compare our method an accelerated version of active contours (AC) to a global convex energy minimization method Chambolle/Pock (CP) [7], and find comparable robustness to global methods but with a significant computational savings. We choose the regularity such that standard active contours converges to a local minima (not the global) over multiple different initializations, so that a better method is required to optimize the energy. The regularity is also chosen with the performance of CP in mind for the comparison, as CP also requires a sufficiently high regularity, although lower than standard active contours, to segment the region.

Table 3.8: [Left]: PDE Acceleration (AC) offers a comparable level of robustness to initialization as global convex Chambolle/Pock (CP) in lower computational time. [Right]: Visual comparison for the results with greatest energy difference in CP & AC shows that the energy differences are nearly in-perceptible.

Near Square	**Time CP (sec)**	**Time AC (sec)**	**Energy AC**	**Energy CP**
Res: 280 x 280	0.176	0.042	5.196E+08	5.186E+08
Res: 560 x 560	2.34	0.11	5.212E+08	5.099E+08
Res: 1120 x 1220	20.009	1.167	5.212E+08	5.073E+08
Res: 2240 x 2240	159.76	14.43	5.202E+08	5.056E+08
Threshold Mask				
Res: 280 x 280	0.174	0.064	5.168E+08	5.225E+08
Res: 560 x 560	2.24	0.204	5.168E+08	5.092E+08
Res: 1120 x 1120	20.438	1.055	5.157E+08	5.071E+08
Res: 2240 x 2240	159.178	14.606	5.201E+08	5.051E+08
Far Square				
Res: 280 x 280	0.771	0.278	5.198E+08	5.186E+08
Res: 560 x 560	8.726	1.43	5.198E+08	5.090E+08
Res: 1120 x 1120	91.197	14.65	5.206E+08	5.069E+08
Res: 2240 x 2240	772.342	69.48	5.201E+08	5.060E+08



We run all AC and CP experiments to convergence and measure the computational time and final energy for 3 different initial segments at 4 different resolutions. The experiments were: a square close to the desired segmentation - “Near square”, a square far from the desired segmentation - “Far square”, and a binary threshold of the image - “Threshold Mask”. In Table 3.8 we present a scaled down noisy binary image of resolution 1120 x 1120 with the final segmentations for both methods. Performance results are provided in Table 3.8. This comparison shows that our method consistently obtains comparable local optima over different initialization, similar to CP, but with less computational time. Furthermore, our method applies more generally to non-convex problems, where we would expect similar robustness in our method, and where CP is not as applicable; see 3D stereoscopic segmentation experiments in Section 4.3.

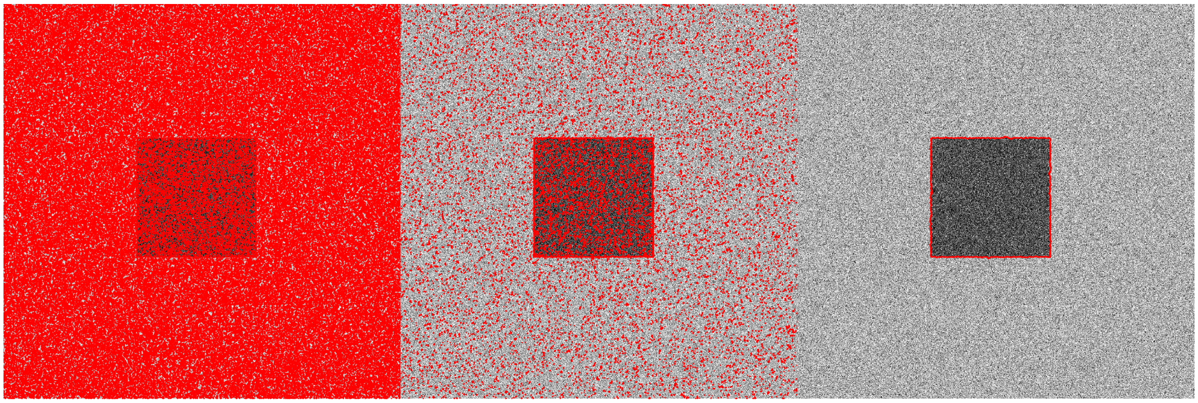


Figure 3.18: [Left]: Initial threshold initialization of noisy square [Middle]: Converged segmentation for Chambolle-Pock. [Right]: Converged segmentation for Accelerated Active Contours.

In Figure Figure 3.18 we present the case of a segmentation based on an initial threshold of the image, this is a similar experiment to the threshold mask discussed earlier, but with a significantly lower arc length penalty than those used in the proceeding table Table 3.8. Here we show another advantage of the Accelerated Active Contours method in needing less regularization for segmentation. In Figure Figure 3.18 the numerical implementation of Chambolle-Pock becomes trapped whereas the accelerated version of active contours is able to push past the local minimizers in the image.

Table 3.9: Performance numbers for Threshold Segmentation Chambolle-Pock vs Accelerated Active Contours for Figure 3.18.

Image Resolution	Arc Penalty	Iterations	CP Energy	AC Energy
1120 x 1120	1.00E+4	600	8.14E+9	8.07E+9

3.5.4 Accelerated Active Surfaces

The accelerated active contour models developed in section 3.5 offer a more robust evolution framework for generic contour based optimization problems, just as the class of Sobolev active contour models introduced earlier. Both methodologies regularize the optimization process, without imposing regularity on the final optimized result, greatly boosting the evolving contour's resistance to spurious or shallow local minimizers. In both cases, this desirable property is achieved by effectively averaging contributions from several local gradient forces in order to determine the instantaneous evolution of any given point on the curve.

In the case of Sobolev active contours, this averaging is done spatially at each fixed time instant by an effective convolution along the curve. Unfortunately, while special tricks exist to do this quickly for closed curves, they do not apply to surfaces or higher dimensional manifolds, where Laplace-Beltrami style PDE's must instead be solved along the surface at every time instant in order to calculate the Sobolev gradeint.

Accelerated active contour models, on the other hand, perform a temporal rather than spatial averaging. As a particle along the curve accelerates, its instantaneous velocity represents the accumulation of local gradient information over its recently traveled trajectory, rather than the accumulation of local gradient information from its neighboring contour points at the same instant in time. An important advantage of the time-based averaging, in contrast with the instantaneous spatially-based averaging in Sobolev style active contours, is that the same computational speed up in 2D will apply equally in 3D and higher.

In the case of geometric active surfaces, we start with a potential energy which depends only upon the geometric shape of the contour S (again, as in the contour case, not its parameterization). Under these assumptions the first variation of the potential energy will have the following form

$$\delta \mathbf{U} = -\int_S f\left(\delta C \cdot N\right) dA$$

where fN represents a force along the unit normal N at each point on the surface S and where dA denotes the surface area measure. The implicit level set framework is particularly convenient for active surfaces given the complexities of dealing with 3D meshes. In the level set framework, the (non-accelerated) gradient descent surface evolution PDE has the same form as in 2D, but is applied to a 3D grid instead. Namely

$$\frac{\partial \psi}{\partial t} = -\hat{f}\|\nabla \psi\|$$

where $\hat{f}(x,t)$ denotes a spatial extension of f to points away from the surface. Narrow band methods are especially important in 3D to keep the computational cost of updating the level set function ψ to a minimum (as well as limiting the neighborhood where extension functions such as $\hat{f}$ need to be computed and evolved).

In the simplest constant density model case, applied to surfaces. the kinetic energy term for the accelerated model will have a similar form but with the density ρ interpreted per unit surface area.

$$\mathbf{T} = \frac{1}{2}\rho \int_S \left(\frac{\partial S}{\partial t} \cdot \frac{\partial S}{\partial t} \right) dA$$

Computing the Euler-Lagrange equation of the generalized action integral (Equation 3.1) and writing it in the level set framework yields the same system of first order PDE's as in the contour case, except now in 3D dimensions,

$$\frac{\partial \psi}{\partial t} = -\hat{\beta}\|\nabla \psi\|, \qquad \frac{\partial \hat{\beta}}{\partial t} = \frac{\lambda k^2 t^{(k-2)}}{\rho}\hat{f} + \nabla \cdot \left(\frac{1}{2}\hat{\beta}^2 \frac{\nabla \psi}{\|\nabla \psi\|} \right) - \frac{k+1}{t}\hat{\beta}$$

where $\hat{f}(x,t)$ and $\hat{\beta}(x,t)$ denote 3D spatial extensions of f and β respectively.

3.6 Stereo Reconstruction

3.6.1 Introduction

The next application of PDE acceleration that we pursued was an extension of stereoscopic segmentation from [13] in which we used a second order accelerated scheme as opposed to their gradient descent update. Consider the smooth surface model from [13]. The authors formulate the energy of their surface as depending on three terms the surface S, the radiance of the surface f and the radiance of the background g. Their cost functional is composed of three integrals: First, they consider a data fidelity term $E_{data}(f, g, S)$ which measures the discrepancy between the measured images and the images predicted by the model. Second, they consider a smoothness term $E_{smooth}(f, g, S)$ which assumes a smooth surface radiance for the surface and background. Third, they consider a geometric prior $E_{geom}(S)$. The combined cost functional is thus constructed as follows:

$$E(f, g, S) = E_{data}(f, g, S) + E_{smooth}(f, g, S) + E_{geom}(S) \tag{3.123}$$

The geometric prior and and smoothness term are given as follows:

$$E_{geom} = \int_S dA \tag{3.124}$$

$$E_{smooth} = \int_S ||\nabla_S f||^2 dA + \int_B ||\nabla g||^2 d\Theta \tag{3.125}$$

which favor a minimal surface area S and a minimization of the quadratic TV norms on the radiance functions f and g. Here ∇_S denotes the intrinsic gradient on the surface S and B is treated as the background with angular coordinates $\Theta = (\theta, \phi)$ which can be mapped one to one with the coordinates $\hat{x_i}$ of each image domain Ω_i through $\Theta = \Theta_i(\hat{x}_i)$. The data penalty term is given as:

$$E_{data}(f, g, s) = \sum_{i=1}^{n} \left(\int_{\Omega_i} f((\pi_i{}^{-1}(\hat{x}_i)) - I_i(\hat{x}_i))^2 d\Omega_i) \right) \tag{3.126}$$

Here we define $x = (x, y, z)$ as representing a generic point of a scene in $\mathbb{R}^3$ which is expressed in global coordinates, based on a fixed inertial reference frame.

$x_i = (x_i, y_i, z_i)$ represents the same point in camera coordinates relative to each image I_i. The stereoscopic segmentation is constrained by a sequence of images I_1 to I_n which is taken from the scene. The authors assume that the domain Ω_i of each image I_i belongs to a 2D plane in which $z_i = 1$ and (x_i, y_i) constitute Cartesian coordinates in this image plane. Furthermore the authors use $\pi_i : \mathbb{R}^3 \to \Omega_i; x \mapsto \hat{x}_i = (\tilde{x}_i, \tilde{y}_i)$ to denote an ideal perspective projection onto the image plane, where $\tilde{x}_i = \frac{x_i}{z_i}$ and $\tilde{y}_i = \frac{y_i}{z_i}$. The authors denote ${\pi_i}^{-1}$ as a back-projection onto the surface S by ${\pi_i}^{-1} : \Omega_i \to S$.

The complete gradient flow for this functional [13] is given as follows:

$$\begin{aligned} \frac{dS}{dt} = \sum_{i=1}^{n} \frac{f-g}{{z_i}^3}[I_i - f + I_i - g](\nabla_i \chi_i \cdot x_i)N + \\ \sum_{i=1}^{n} \frac{2\chi_i}{{z_i}^3}(I_i - f)(\nabla_i f \cdot x_i)N - HN + \\ K\langle \nabla_S f, A^{-1}\nabla_S f\rangle N - ||\nabla_S f||HN \end{aligned} \tag{3.127}$$

where H denotes mean curvature, N the outward unit normal, K the Gaussian curvature of S, and A denotes the second fundamental form of S with respect to the isothermal coordinates. The authors denote χ_i as a special characteristic function where $\chi_i(x) \in \{0, 1\}$ and $\chi_i(x) = 1$ for $x \in {\pi_i}^{-1}(\Omega_i)$ and $\chi_i(x) = 0$ for $x \notin {\pi_i}^{-1}(\Omega_i)$. This is performed so as to exclude points that are occluded by other points on the surface S [9].

Although there are simpler models for shape reconstruction [116, 88] we choose this model [13] because of its extreme non-convexity. The reason for this non-convexity is that the projections in stereoscopic segmentation must be applied to both the occlusion boundaries and the evolving structure. The gradient descent optimization used in this stereoscopic segmentation model is also prone to getting trapped in local

[9] We would refer the reader to [13, 129] for the details of deriving the gradient descent equations.

minima and demonstrates the potential benefits of PDE acceleration.

3.6.2 Simulated Results Gradient Descent vs PDE Acceleration

One of the drawbacks of the stereoscopic segmentation presented in [13] is that it is unable to resolve sufficiently large local minimizers [116]. To visualize this we illustrate the following experiment:

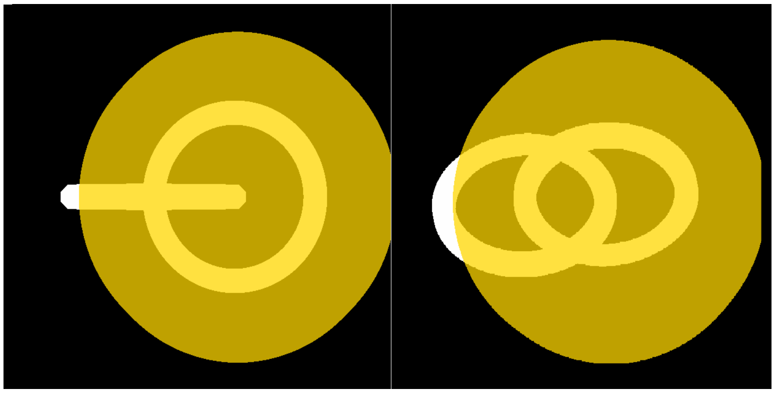


Figure 3.19: Projected silhouettes (in yellow) from starting surface which does not intersect tori holes.

Consider a pair of double tori in which the interior holes are completely enveloped by the initial ellipsoid see Figure 3.19. Using stereoscopic segmentation we run the process until convergence at which point we are trapped see Figure 3.20 and unable to carve out the interior regions of the tori.

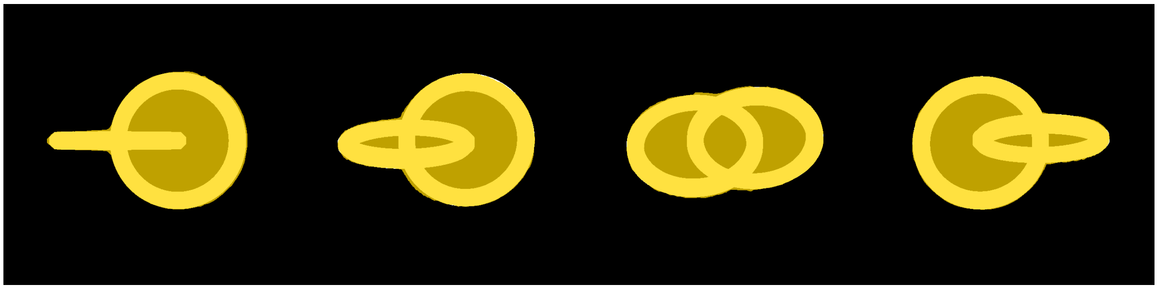


Figure 3.20: Converged 3D Silhouettes of Gradient Descent

With the older technique there are some limited improvements we can make such as by using more views, and a much stronger area penalty. In Figure 3.21 we show the initial result with 16 images and no area penalty [Left]. Here the gradient-descent clearly gets trapped in a local minima and is unable to push through the interiors. While adding additional camera views and some regularity does improve the results, [Middle] image, the large local minimizers still remain. Adding a much harsher area penalty can remove the local minimizers, however, it does so at the expense of image features. With delicately and deliberately chosen over-regularization, gradient descent can thus penetrate the tori holes but at the expense of breaking the tori rings themselves (see right image of Figure 3.21).

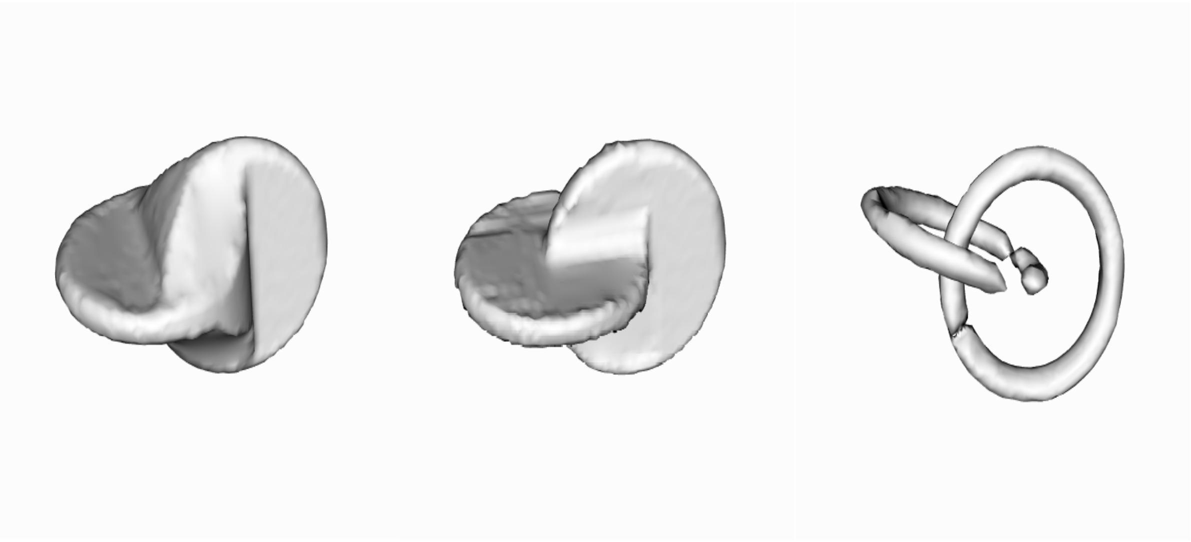


Figure 3.21: Three double tori reconstructions using gradient-descent stereoscopic segmentation. [Left] torus used 16 images and 0 area penalty. [Middle] torus used 18 images and a moderate area penalty of 150. [Right] torus used 18 images and a high area penalty of 500. Notice the modest feature loss in the final image.

A similar experiment by Kolev, Brox, and Cremers [116] compares a double torus reconstruction using stereoscopic segmentation against a convex probabilistic fusion scheme that combines probable 3D shapes and observed color information. In [116] the authors astutely note that while the approximation in [129, 13] is more faithful it is not globally optimizable. However, by extending the non-convex the approach in [13] using PDE acceleration we are able to push through the local minimizers without the use of additional constraints on the geometry or even additional regularization.

Our objective was that by using PDE acceleration here we would be able to circumvent that limitation. To this end we recreated a synthetic torus experiment that mimicked the conditions of the experiment in [116]. The initial enclosing ellipsoid (see Figure 3.19) completely covered both tori centers. When the gradient descent stereoscopic segmentation executed it was only able to partially resolve the shape of the tori; see Figure 3.20 and Figure 3.22. While the accelerated method was able to converge faster, the primary benefit of acceleration in this experiment is that it can reach a better local minimum in the reconstruction when compared with gradient descent. For acceleration the momentum allowed it to push into the interiors of the tori.

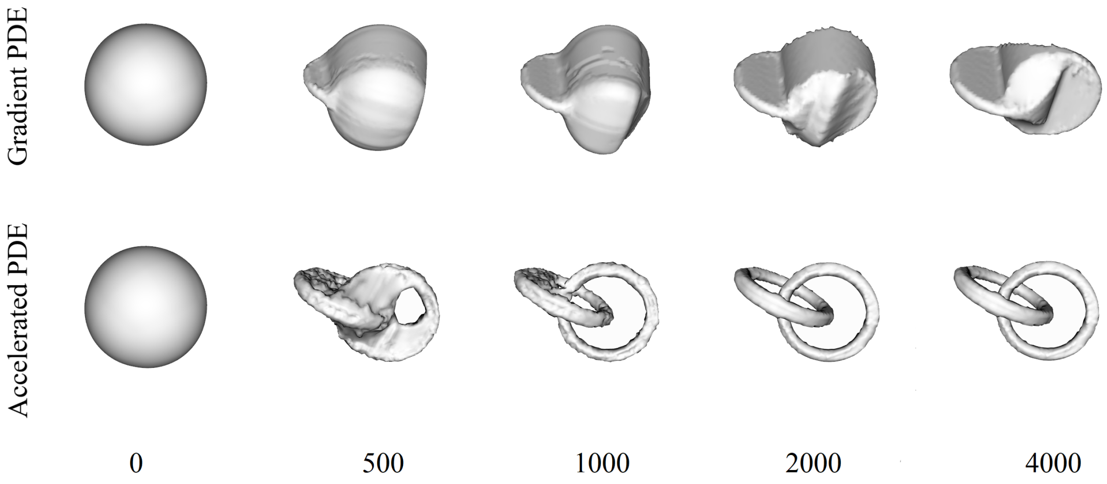


Figure 3.22: Comparison of Stereoscopic Segmentation on synthetic data for Gradient PDE method vs Accelerated PDE method at different iteration steps 0, 500, 1000, 2000 and 4000 iterations respectively. The gradient method gets trapped by local minimizers in this case the bounds of the torus and is unable to form the holes without heavy regularization. Note that the initial start for the both methods is the ellipsoid. The area penalty was fixed at 0 and the experiment used 16 image views of the tori for the reconstruction.

A numeric performance comparison between the two methods is provided in Table 3.10.

Table 3.10: Performance benchmark of gradient vs accelerated method, computations were done on an intel 6 Core i7-5930K.

	Iterations to Converge:	Run Time (sec):	Final Energy
Gradient Descent	3653	196.44	2.09E+10
PDE Acceleration	1630	66.16	2.90E+09

3.6.3 Physical Calibration Array and Real Data

The second reconstruction we present is of a wooden horse using 32 different 640 x 480 resolution images; see Figure Figure 3.23. Comparing the Gradient Descent [Bottom] and Accelerated PDE [Top] schemes, we demonstrate faster convergence for the Accelerated PDE scheme as well as better robustness to local minimizers. Unlike in the Gradient PDE method we are able to capture the entire horse volume and do not lose smaller features such as the horse's thin legs.

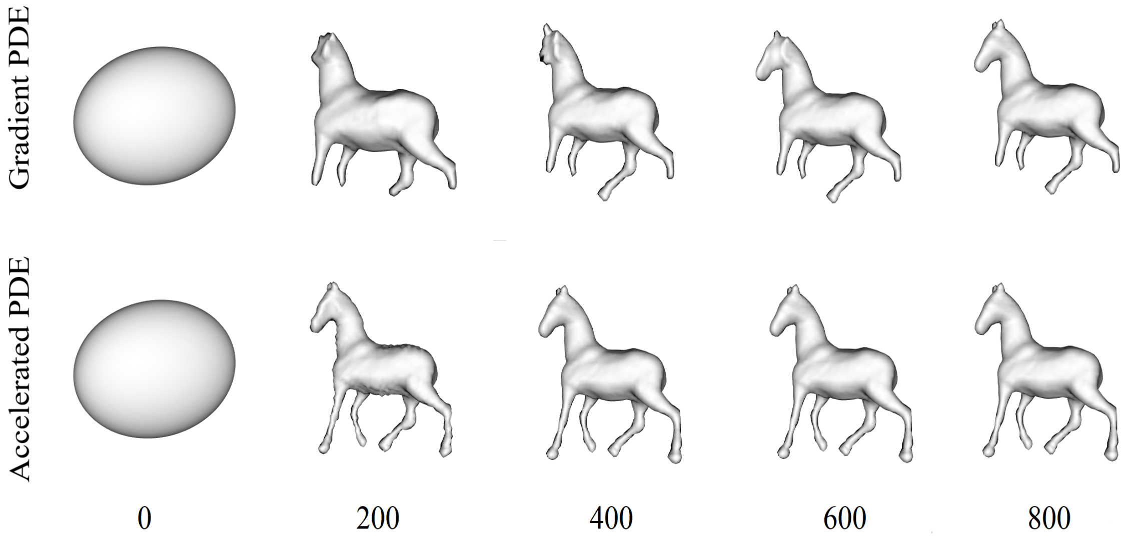


Figure 3.23: [Top]: Gradient descent driven variational 3D reconstruction gets trapped in local minima when strongly regularized [Bottom]: PDE acceleration is able to successfully shoot past local minimizers under a higher area penalty. Note average convergence time for PDE acceleration is almost three times faster than Gradient descent. Iterations are given at the bottom with performance results found in table Table 3.11.

Imposing a high area penalty on the horse reconstruction creates local minimizers within the horse's narrow legs. Gradient descent stereoscopic segmentation then struggles to push down the legs and reconstruct the entire horse volume. The accelerated version progresses far faster and does not suffer a penalty even under the same heavy regularization.

Table 3.11: Performance numbers for Gradient Descent vs PDE Acceleration of variational 3D reconstruction. Note that increase in energy is due to the methods not capturing the entire horse volume.

		Gradient Descent			Acceleration	
Area Penalty	Iterations	Run Time (sec)	Final Energy	Iterations	Run Time (sec)	Final Energy
300	751	27.00	1.099E+10	205	8.307	9.381E+09

After we obtained promising results in our reconstructed simulations we then moved to the next phase of the research where I along with another PhD student, Huizong Yang, prototyped and built a 3D scanning array for doing 3D reconstruction. Since the accelerated method is more robust and needs fewer camera angles to work we could reduce the number of cameras in the initial array.

Currently, we have finished the calibration of the array and have begun performing new experiments. The next extension that we hope to add, although this would be primarily Huizong's focus would be the addition of depth sense cameras to the array. This would allow us to incorporate depth information into the energy functional and to potentially resolve much finer surface features.

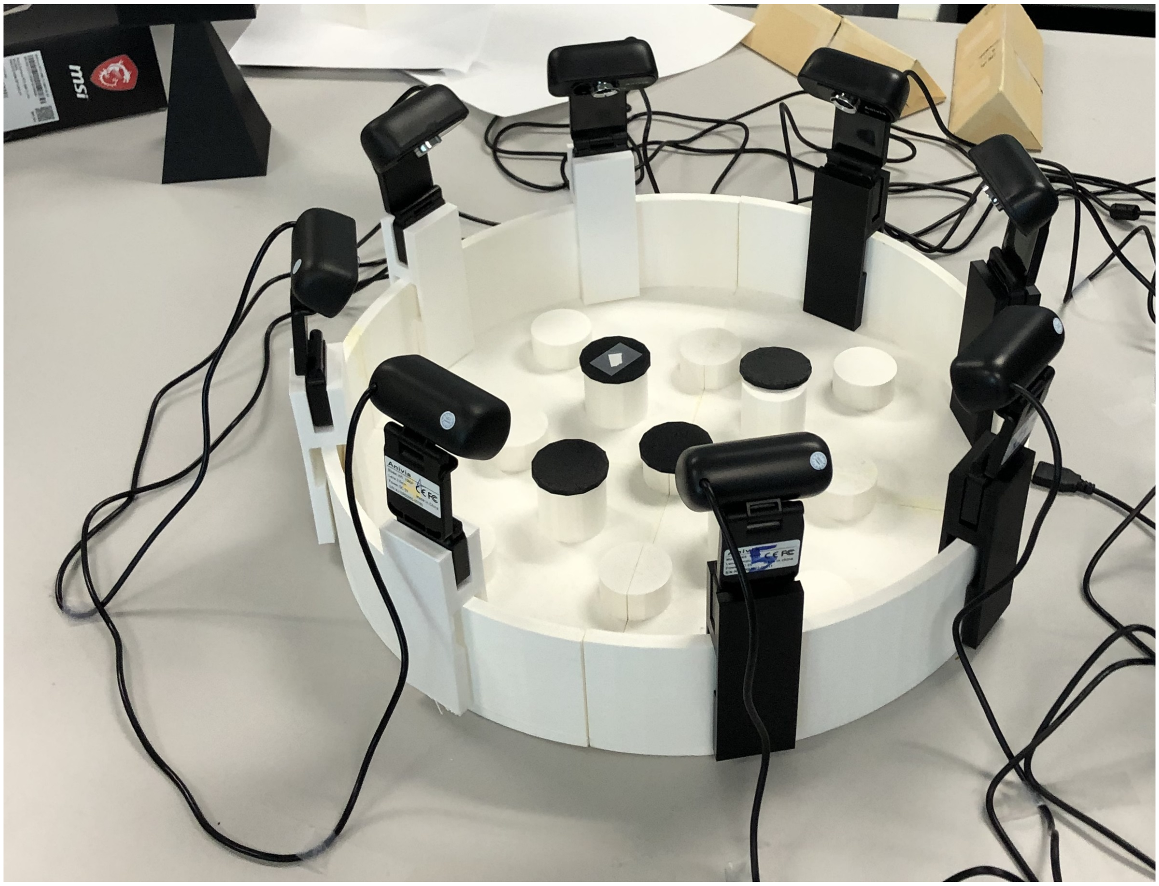


Figure 3.24: Stereo Reconstruction Camera Array

CHAPTER 4
CONCLUSION

In this dissertation we derived the PDE acceleration framework and demonstrated its utility for a wide ranging class of computer vision problems and as a general method for solving variational PDEs. PDE acceleration showed extensive benefits for regularized inversion problems, optical flow, active contour driven segmentation and even highly non-convex formulations such as stereoscopic-segmentation which we used in 3D reconstruction.

4.0.1 Discretization Schemes

In our research on PDE acceleration we first showed how the accelerated gradient is generalized from the Bregman Lagrangian in the ODE domain to the much more general PDE setting. Our research then examined how to best implement discretizations of the accelerated gradient method. We reviewed several discretization schemes along with their accompanying stability conditions (see section 3.2). We further demonstrated how the framework can be easily and quickly adopted for existing variational formulations.

4.0.2 Inversion Problems

In our experiments with regularized inversion problems we began with the case of simple denoising. We showed how PDE acceleration had comparable or better performance than methods such as Split Bregman and Primal Dual. We also demonstrated how to choose the optimal damping coefficient and illustrated the performance differences of varying damping schemes. In our later experiments with image deconvolution we demonstrated how the accelerated descent method could outperform both

Chambolle-Pock Primal Dual and ADMM.

Compressed Sensing

Another successful application of the PDE acceleration framework was for compressed sensing in particular edge-preserving TV. As in the earlier experiments with deconvolution we expected and saw a large performance uplift over the current methods (see Figure 3.9) with the additional advantage that we would not have to convexify the energy functional, which is necessary for the current dual formulation of the problem.

Poisson Solver

An incredibly useful application of PDE acceleration is as a more general optimization method for solving PDEs. In this text we showed how to undertake this for Poisson's equation, which is broadly utilized in physics. Traditional methods for solving Poisson's equation have focused on either relaxation schemes such as Gauss-Seidel, the Jacobi method, and SOR or matrix methods like conjugate gradient [130]. The accelerated descent scheme was shown to be quite competitive against the standard optimizers (see Table 3.5) and was orders of magnitude better than the Jacobi method. In problems with large matrices (see Table 3.6) this benefit became even more pronounced as PDE acceleration was nearly three times faster than SOR.

4.0.3 Optical Flow

The initial optical flow experiments demonstrated an order of magnitude improvement in speed over the conjugate gradient method (see Table 3.7). Both methods converged to the same local minimum (see Figure 3.14) but the accelerated optical flow was more robust to lack of initialization and converged in far less time. The initial exploration into optical flow was done with the aim of testing the effectiveness of the accelerated scheme against one of the most common methods conjugate gradient.

4.0.4 Active Contours

One of the other contributions of this research was to extend PDE acceleration into the active contour domain. In active contour driven segmentation previous methods would often get trapped in local minimizers due to noise (see Figure 3.15). However, by reformulating the problem as a convex optimization, researchers could leverage global optimization methods such as primal dual as was done in our experiments (see Table 3.8). We show that with PDE acceleration we can achieve better results by not relying on the convexification of the problem and instead using a more faithful representation of the cost functional. In addition we showed a numerical benefit of reduced reliance on regularity and were able to better capture the segmented squares more accurately (see Table 3.8).

4.0.5 Stereo Reconstruction

In the original stereoscopic segmentation paper [13], one of the primary drawbacks of the 3D reconstruction method was an inability to handle local minimizers in the silhouette geometry[1]. In response to work such as [116] where the authors rightly point out the limitations of gradient-descent driven stereoscopic segmentation we have extended the PDE acceleration framework to the formulation in [13]. In subsection 3.6.2, we present an experiment featuring a synthetic data set that closely mirrors the synthetic stereo data[2] from [116]. In this experiment we demonstrate one of the principle advantages of pde acceleration in that we are not only able to improve convergence times as was shown in the example of optical flow but that we are often able to resolve to far better local minimizers than what might be obtained using gradient descent. We further demonstrate on real data (see Figure 3.23), how PDE acceleration has both a higher robustness to over regularization and a much

[1] See subsection 3.6.2 and [16]

[2] The data set and code from [116] was not available and thus had to be re-implemented.

faster convergence rate in highly non-convex applications.

4.0.6 Future Work

A quite natural extension of this initial work in PDE acceleration would be to derive more of the theory required to optimally predict the tuning parameters that are used particularly the damping. In a recent paper by Calder and Yezzi [34] the authors were able to derive an optimal formula for the damping coefficient for a subset of PDE accelerated functionals using an analysis of the boundary value conditions. What we have found in the course of my research is that the approach in [34] does not necessarily generalize well to other functionals such as those used in deconvolution and stereoscopic-segmentation. Therefore we are of the opinion that further theory is needed to develop this portion of the framework.

Appendices

APPENDIX A
FACILITY ACKNOWLEDGEMENT

The majority of the work for my PhD research was performed on my two personal computers as well as on the lab machine at Tech Square Research Building (TSRB).

I have also made use of the TSRB work space to build and design a 3D scanning array which was built in partnership with fellow PhD student Huizong Yang. For this array we used the 3D printing facilities located in Van Leer and Georgia Tech's Hive lab to print the structure and requisite calibration pattern.

APPENDIX B
EXPERIMENTAL EQUIPMENT

The experimental equipment used in the fulfillment of this research were the lab computers provided by Professor Yezzi as well as my own personal machines. I also in conjunction with fellow PhD student Huizong Yang designed and built a stereo rig for stereo reconstruction. This rig makes use of 8 different cameras.

APPENDIX C
DATA PROCESSING

The deconvolution of the blurred butterfly was initially processed as a symmetric image. One of the issues that our research group observed but was unable to amend in time for the initial Journal of Mathematical Imaging and Vision (JMIV) paper [122] was the issue of severe ringing caused by repeated 2D FFTs on non-symmetric images. This ringing was amplified by the 2nd order scheme that we used in the deconvolution of these images. Thus all of the routines were fed a symmetric image that was 4 times the initial resolution of the blurred butterfly. The routines for this experiment were written in MATLAB.

The experiments with the Poisson equation and compressed sensing were not yet published at the time this dissertation was written. Thus these experiments might change in future publications of this work.

The second experiment for optical flow used images from the middlebury optical flow data set. In the traditional optical flow benchmarks the ground truth is unknown and the algorithms are tested on an untrained/untuned data set. Since the intent of the experiment was not to achieve stellar results in optical flow but merely to show a significant run time improvement a better optical flow model could have been used.

Both the conjugate gradient method and the 2nd order accelerated descent scheme arrived at the same local minimum but took vastly different paths to get there. The conjugate gradient method is also somewhat finicky and the images must be carefully down sampled when initiating the pyramid scheme. A poor initialization of the conjugate gradient method strongly hampers performance when compared to the accelerated scheme. The routines for this set of experiments were written in Python.

The third experiment on Stereo Reconstruction was based on a 2011 paper by

Daniel Cremers. In his paper Cremers implemented a synthetic experiment where he compared the stereoscopic segmentation from [13] against his convex formulation. The experiment in subsection 3.6.2 is an extension of the initial work done by Yezzi where a second order scheme is used in place of gradient descent. The experiment used Opengl to create an adjustable pair of tori. The artificial camera calibrations were then saved and used to reconstruct the pair of double tori. The experiment was slightly different from Cremers but achieved a visually similar local minimum for the gradient descent case. The routines for this set of experiments were written in C++ and Opengl.

APPENDIX D
PUBLISHED CODES

To encourage reproducibility and citation of this and other manuscripts, the software implementations for a number of these experiments are provided as GitHub repositories. The associated links for the experiments and provided code are given as follows:

- Accelerated (ACC) Denoising: https://github.com/minasbenyamin/Denoising-
- ACC Deconvolution: https://github.com/minasbenyamin/Deconvolution
- Optical Flow: https://github.com/minasbenyamin/Middlebury
- Optical Flow Translation: https://github.com/minasbenyamin/opt_flow_translation
- Active Contours: https://github.com/minasbenyamin/Active-Contours-
- Other Published Codes: https://github.com/minasbenyamin/DissertationCodes

The remaining codes are still undergoing code revision and or peer review. Links to these repositories will eventually be posted to the Readme file of the *Other Published Codes* repository. The programming languages used in the preparation of this dissertation are MATLAB, Python and C++. Performance results may vary based on software library versions and computing hardware. Please pay close attention to the Readme documentation as it will detail necessary publications to cite when reproducing/comparing this work.

APPENDIX E
ALTERNATIVE DISCRETIZATION SCHEMES FOR BELTRAMI REGULARIZERS

In this section of the appendix we summarize the iterative schemes for the deconvolution problem with a Beltrami norm. Note that all of our experiments with deconvolution used the *2-order Accelerated* scheme and that $D^2_{\beta,\Delta x}u^n$ denotes a particular discretization (see subsection 3.3.3) of the Beltrami term.

Gradient Descent

$$\begin{aligned} \Delta u^n = -\Delta t \left(\lambda K^T * (K * u^n - g) - D^2_{\beta,\Delta x}u^n\right) \\ {\scriptstyle \Delta t \leq \Delta x^2 \left(\frac{2}{4N\beta + \lambda\Delta x^2}\right)} \end{aligned} \tag{E.1}$$

Beginning with the gradient descent update, we define the terms of this equation as follows: n is the current iteration, Δu^n refers to the update to the evolving entity u^n, Δt is the time step, λ is the fidelity coefficient on the data term, K is the convolution operator, and g is the initial value of evolving entity in the problem. For the time constraint: β is the coefficient in the Beltrami norm and Δx refers to the step size of the N-dimensional Cartesian grid used to approximate the spatial derivatives of the Laplacian term $D^2_{\beta,\Delta x}u^n$. Moving to the accelerated schemes (Equation E.2 and Equation E.3) we define a as the damping coefficient used in the PDE acceleration

1-order Accelerated

$$\begin{aligned} \Delta u^n = \frac{1}{1 + a\Delta t}\Delta u^{n-1} - \frac{\Delta t^2}{1 + a\Delta t}\left(\lambda K^T * (K * u^n - g) - D^2_{\beta,\Delta x}u^n\right) \\ {\scriptstyle \Delta t \leq \Delta x \left(\sqrt{\frac{4}{4N\beta + \lambda\Delta x^2} + \left(\frac{a\Delta x}{4N\beta + \lambda\Delta x^2}\right)^2} + \frac{a\Delta x}{4N\beta + \lambda\Delta x^2}\right)} \end{aligned} \tag{E.2}$$

and Δu^{n-1} as the previous update to the evolving entity.

2-order Accelerated

$$\begin{aligned}\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t}\Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t}\left(\lambda K^T * (K * u^n - g) - D^2_{\beta,\Delta x} u^n\right) \\ {\scriptstyle \Delta t \leq \Delta x \left(\frac{2}{\sqrt{4N\beta + \lambda\Delta x^2}}\right)}\end{aligned} \tag{E.3}$$

For the remaining implicit scheme v^n and u^n are updated sequentially and v^n is used as an intermediate variable for computing u^{n+1}.

Semi-Implicit

$$\begin{aligned}v^n &= u^n + \frac{2 - a\Delta t}{2 + a\Delta t}\Delta u^{n-1} \\ u^{n+1} = v^n - \frac{2\Delta t^2}{2 + a\Delta t} &\left(\lambda K^T * (K * v^n - g) - D^2_{\beta,\Delta x} v^n\right) \\ &{\scriptstyle \Delta t \leq \Delta x \left(\frac{2}{\sqrt{3\left(4N\beta + \lambda\Delta x^2\right)}}\right)}\end{aligned} \tag{E.4}$$

Of these initial schemes we observed that the *2-order Accelerated* update had the most straightforward stability condition and that the *1-order* and *2-order* schemes were both faster than the *Semi-Implicit* method.

APPENDIX F
DISCRETIZATION SCHEMES FOR TOTAL VARIATION REGULARIZATION

In this section of the appendix we summarize the iterative schemes for the deconvolution problem with a TV norm. Like in the proceeding section our experiments were carried out using the *2-order Accelerated* scheme and $D^2_{\Delta x}u^n$ denotes a particular discretization (see subsection 3.3.3) of the TV term.

Gradient Descent

$$\Delta u^n = -\Delta t \left(\lambda K^T * (K * u^n - g) - D^2_{\Delta x}u^n\right) \tag{F.1}$$
$$\Delta t \leq Q\Delta x \left(\frac{2}{4\sqrt{N}+\lambda Q\Delta x}\right)$$

Beginning with the gradient descent update, the terms are defined almost exactly as in Appendix E. The main difference is that there is no β term for the TV norm and that Q is the quantization interval between neighboring pixels. Moving to the accelerated schemes (Equation F.2 and Equation F.3) we again define a as the damping coefficient used in the PDE acceleration and Δu^{n-1} as the previous update to the evolving entity.

1-order Accelerated

$$\Delta u^n = \frac{1}{1+a\Delta t}\Delta u^{n-1} - \frac{\Delta t^2}{1+a\Delta t}\left(\lambda K^T * (K * u^n - g) - D^2_{\Delta x}u^n\right) \tag{F.2}$$
$$\Delta t \leq \sqrt{Q\Delta x}\left(\sqrt{\frac{4}{4\sqrt{N}+\lambda Q\Delta x}+\left(\frac{a\sqrt{Q\Delta x}}{4\sqrt{N}+\lambda Q\Delta x}\right)^2}+\frac{a\sqrt{Q\Delta x}}{4\sqrt{N}+\lambda Q\Delta x}\right)$$

For the remaining implicit scheme (Equation F.4) v^n and u^n are again updated sequentially and v^n is used as an intermediate variable for computing u^{n+1}.

2-order Accelerated

$$\Delta u^n = \frac{2 - a\Delta t}{2 + a\Delta t}\Delta u^{n-1} - \frac{2\Delta t^2}{2 + a\Delta t}\left(\lambda K^T * (K * u^n - g) - D^2_{\Delta x} u^n\right) \tag{F.3}$$
$$\scriptstyle \Delta t \leq \sqrt{Q\Delta x}\left(\sqrt{\frac{4}{4\sqrt{N}+\lambda Q\Delta x}}\right)$$

Semi-Implicit

$$\begin{aligned} v^n &= u^n + \frac{2 - a\Delta t}{2 + a\Delta t}\Delta u^{n-1} \\ u^{n+1} &= v^n - \frac{2\Delta t^2}{2 + a\Delta t}\left(\lambda K^T * (K * v^n - g) - D^2_{\Delta x} v^n\right) \end{aligned} \tag{F.4}$$
$$\scriptstyle \Delta t \leq \sqrt{Q\Delta x}\left(\sqrt{\frac{4}{3\left(4\sqrt{N}+\lambda Q\Delta x\right)}}\right)$$

As was the case with the Beltrami norm we observed that the *1-order* and *2-order* schemes were also both faster than the *Semi-Implicit* method.

APPENDIX G

ENERGY GRADIENT FOR OPTICAL FLOW

In this section of the appendix we derive the gradient of the energy for optical flow. We begin by stating the relationship between the functional U, which is the energy of our cost functional, with respect to a mapping of the brightness displacement ϕ.

Definition 1 (Functional Gradients) *Let $U : \mathit{Diff}(\mathbb{R}^n) \to \mathbb{R}$. The gradient (or functional derivative) of U with respect to $\phi \in \mathit{Diff}(\mathbb{R}^n)$, denoted $\nabla U(\phi)$, is defined as the $\nabla U(\phi) \in T_\phi \mathit{Diff}(\mathbb{R}^n)$ that satisfies*

$$\delta U(\phi) \cdot v = \int_{\phi(\mathbb{R}^n)} \nabla U(\phi)(x) \cdot v(x) \, \mathrm{d}x \tag{G.1}$$

for all $v \in T_\phi \mathit{Diff}(\mathbb{R}^n)$. The left hand side is the directional derivative and is defined as

$$\delta U(\phi) \cdot v := \left. \frac{\mathrm{d}}{\mathrm{d}\varepsilon} U(\phi + \varepsilon v) \right|_{\varepsilon = 0} . \tag{G.2}$$

Note that $(\phi + \varepsilon v)(x) = \phi(x) + \varepsilon v(\phi(x))$ for $x \in \mathbb{R}^n$.

We now show the computation of the gradient for the illustrative potential (Equation 3.106) used in this paper. First, let us consider the data term $U_1(\phi) = \int_{\mathbb{R}^n} |I_1(\phi(x)) - I_0(x)|^2 \, \mathrm{d}x$ then

$$\delta U_1(\phi) \cdot \delta\phi = \int_{\mathbb{R}^n} 2(I_1(\phi(x)) - I_0(x)) DI_1(\phi(x)) \widehat{\delta\phi}(x) \, \mathrm{d}x = \\ \int_{\phi(\mathbb{R}^n)} 2(I_1(x) - I_0(\psi(x))) DI_1(x) \delta\phi(x) \det \nabla\psi(x) \, \mathrm{d}x,$$

where $\widehat{\delta\phi} = \delta\phi \circ \phi$, $\psi = \phi^{-1}$ and we have performed a change of variables.

Thus, $\nabla U_1 = 2\nabla I_1(I_1 - I_0 \circ \psi) \det \nabla\psi$. Now consider the regularity term $U_2(\phi) = \int_{\mathbb{R}^n} |\nabla(\phi(x) - x)|^2 \, \mathrm{d}x$, then

$$\delta U(\phi) = 2\int_{\mathbb{R}^n} \operatorname{tr}\left(\nabla(\phi(x) - \mathrm{id})^T \nabla\widehat{\delta\phi}(x)\right) \, \mathrm{d}x =$$
$$-\int_{\mathbb{R}^n} \Delta\phi(x)^T \delta\phi(x) \, \mathrm{d}x = \int_\Omega (\Delta\phi)(\psi(x))^T \delta\phi(x) \det \nabla\psi(x) \, \mathrm{d}x.$$

Note that in integration by parts, the boundary term vanishes since we assume that $\phi(x) = x$ as $|x| \to \infty$. Thus, $\nabla U_2 = (\Delta\phi) \circ \psi \det \nabla\psi$.

APPENDIX H

DERIVATIONS FOR ACTIVE CONTOURS

In this section of the appendix we derive the PDE acceleration equations for the active contour case. Here we consider the case of constant density and provide the derivations for Equation 3.116, Equation 3.120 and Equation 3.121.

calculation of Equation 3.116

We begin by restating Equation 3.115:

$$\frac{\partial C}{\partial t} = \alpha T + \beta$$

where α and β denote the tangential and normal speeds of the evolving curve $C(p,t)$, t represents the evolution parameter, and $p \in [0,1]$ denotes an independent parameter along each fixed curve. The unit tangent, unit normal, and curvature are again denoted by $T = \frac{\partial C}{\partial s}$, N and κ respectively. Differentiating (Equation 3.115) with respect to the arclength parameter s yields:

$$\frac{\partial^2 C}{\partial s \partial t} = \frac{\partial \alpha}{\partial s} T + \alpha \underbrace{\frac{\partial T}{\partial s}}_{\kappa N} + \frac{\partial \beta}{\partial s} N + \beta \underbrace{\frac{\partial N}{\partial s}}_{-\kappa T} = \left(\frac{\partial \alpha}{\partial s} - \beta \kappa \right) T + \left(\frac{\partial \beta}{\partial s} + \alpha \kappa \right) N$$

and differentiating $T = \frac{\partial C}{\partial s}$ yields:

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial t}\frac{\partial C}{\partial s} = \frac{\partial}{\partial t}\left(\frac{\frac{\partial C}{\partial p}}{\left\| \frac{\partial C}{\partial p} \right\|} \right) = \frac{\frac{\partial^2 C}{\partial t \partial p}}{\left\| \frac{\partial C}{\partial p} \right\|} - \frac{\frac{\partial C}{\partial p}}{\left\| \frac{\partial C}{\partial p} \right\|^2} \frac{\partial}{\partial t} \left\| \frac{\partial C}{\partial p} \right\| = \frac{\frac{\partial^2 C}{\partial p \partial t}}{\left\| \frac{\partial C}{\partial p} \right\|} - \frac{\frac{\partial C}{\partial p}}{\left\| \frac{\partial C}{\partial p} \right\|^2} \frac{\frac{\partial^2 C}{\partial p \partial t} \cdot \frac{\partial C}{\partial p}}{\left\| \frac{\partial C}{\partial p} \right\|}$$

$$= \frac{\partial^2 C}{\partial s \partial t} - T \left(\frac{\partial^2 C}{\partial s \partial t} \cdot T \right) = \left(\frac{\partial^2 C}{\partial s \partial t} \cdot N \right) N = \left(\frac{\partial \beta}{\partial s} + \alpha \kappa \right) N$$

which gives the first part of Equation 3.116 with the second part due to the rotation relationship between T and N.

CALCULATION OF EQUATION 3.120

Recall the form of the time-explicit generalized action integral:

$$\int \frac{t^{k+1}}{k}\left(\mathbf{T} - \lambda k^2 t^{k-2}\mathbf{U}\right)\, dt \tag{H.1}$$

Let $C(p,t)$ denote a parameterization of the evolving curve C with a time-independent spatial parameter p and with s denoting the time-dependent arclength parameter that we compute. Ignoring the temporary boundary terms, apply integration by parts and assume a closed curve so that the spatial boundary terms cancel:

$$\begin{aligned}
&\delta \int_0^1 \frac{t^{k+1}}{k}\left(\mathbf{T} - \lambda k^2 t^{k-2}\mathbf{U}\right)\, dt = \delta \int_0^1 \left(\int_0^1 \frac{1}{2}\frac{t^{k+1}}{k}\rho\frac{\partial C}{\partial t}\cdot\frac{\partial C}{\partial t}\, ds - \lambda k t^{2k-1}\mathbf{U}\right)\, dt = \int_0^1 \delta\left(\int_0^1 \frac{1}{2}\frac{t^{k+1}}{k}\rho\frac{\partial C}{\partial t}\cdot\frac{\partial C}{\partial t}\left\|\frac{\partial C}{\partial p}\right\| dp - \lambda k t^{2k-1}\mathbf{U}\right)\, dt \\
&= \int_0^1 \frac{\rho}{k}\left(\int_0^1 t^{k+1}\left(\frac{\partial C}{\partial t}\cdot\delta\frac{\partial C}{\partial t}\left\|\frac{\partial C}{\partial p}\right\| + \frac{1}{2}\left\|\frac{\partial C}{\partial t}\right\|^2\delta\left\|\frac{\partial C}{\partial p}\right\|\right) dp - \frac{\lambda k^2 t^{2k-1}}{\rho}\delta\mathbf{U}\right)\, dt \\
&= \int_0^1 \frac{\rho}{k}\left(\int_0^1 -\frac{\partial}{\partial t}\left(t^{k+1}\frac{\partial C}{\partial t}\left\|\frac{\partial C}{\partial p}\right\|\right)\cdot\delta C + \frac{1}{2}t^{k+1}\left\|\frac{\partial C}{\partial t}\right\|^2\delta\frac{\partial C}{\partial p}\cdot\frac{\partial C}{\partial s}\, dp + \frac{\lambda k^2 t^{2k-1}}{\rho}\int_C f\left(\delta C\cdot N\right) ds\right)\, dt \\
&= \int_0^1 \frac{\rho}{k}\left(\int_0^1 \left(-\,t^{k+1}\frac{\partial^2 C}{\partial t^2}\left\|\frac{\partial C}{\partial p}\right\| - (k+1)t^k\frac{\partial C}{\partial t}\left\|\frac{\partial C}{\partial p}\right\| - t^{k+1}\frac{\partial C}{\partial t}\frac{\partial}{\partial t}\left\|\frac{\partial C}{\partial p}\right\| - \frac{1}{2}t^{k+1}\frac{\partial}{\partial p}\left(\left\|\frac{\partial C}{\partial t}\right\|^2\frac{\partial C}{\partial s}\right)\right)\cdot\delta C\, dp + \frac{\lambda k^2 t^{2k-1}}{\rho}\int_C fN\cdot\delta C\, ds\right)\, d \\
&= \int_0^1 \frac{t^{k+1}}{k}\rho\left(\int_0^1 \left(-\frac{\partial^2 C}{\partial t^2}\left\|\frac{\partial C}{\partial p}\right\| - \frac{k+1}{t}\frac{\partial C}{\partial t}\left\|\frac{\partial C}{\partial p}\right\| - \frac{\partial C}{\partial t}\left(\frac{\partial^2 C}{\partial p\partial t}\cdot\frac{\partial C}{\partial s}\right) - \frac{1}{2}\frac{\partial}{\partial p}\left(\left\|\frac{\partial C}{\partial t}\right\|^2\frac{\partial C}{\partial s}\right)\right)\cdot\delta C\, dp + \frac{\lambda k^2 t^{k-2}}{\rho}\int_C fN\cdot\delta C\, ds\right)\, dt \\
&= \int_0^1 \frac{t^{k+1}}{k}\rho\int_C \underbrace{\left(-\frac{\partial^2 C}{\partial t^2} - \frac{k+1}{t}\frac{\partial C}{\partial t} - \left(\frac{\partial^2 C}{\partial s\partial t}\cdot\frac{\partial C}{\partial s}\right)\frac{\partial C}{\partial t} - \frac{\partial}{\partial s}\left(\frac{1}{2}\left\|\frac{\partial C}{\partial t}\right\|^2\frac{\partial C}{\partial s}\right) + \frac{\lambda k^2 t^{k-2}}{\rho}fN\right)}_{\text{Set to zero for Euler-Lagrange equation}}\cdot\delta C\, ds\, dt
\end{aligned}$$

CALCULATION OF EQUATION 3.121

Decomposing the acceleration C_{tt} into tangential and normal components yields:

$$\frac{\partial^2 C}{\partial t^2} = -\frac{k+1}{t}\frac{\partial C}{\partial t} - \left(\frac{\partial^2 C}{\partial s \partial t} \cdot \frac{\partial C}{\partial s}\right)\frac{\partial C}{\partial t} - \left(\frac{\partial^2 C}{\partial s \partial t} \cdot \frac{\partial C}{\partial t}\right)\frac{\partial C}{\partial s} - \frac{1}{2}\left\|\frac{\partial C}{\partial t}\right\|^2 \frac{\partial^2 C}{\partial s^2} + \frac{\lambda k^2 t^{k-2} f}{\rho} N$$

$$= -\frac{k+1}{t}\frac{\partial C}{\partial t} - \left(\frac{\partial \alpha}{\partial s} - \beta\kappa\right)\frac{\partial C}{\partial t} - \left(\alpha\frac{\partial \alpha}{\partial s} + \beta\frac{\partial \beta}{\partial s}\right)T - \frac{\alpha^2+\beta^2}{2}\kappa N + \frac{\lambda k^2 t^{k-2} f}{\rho} N$$

$$\frac{\partial^2 C}{\partial t^2} \cdot T = -\left(\frac{k+1}{t} + \frac{\partial \alpha}{\partial s} - \beta\kappa\right)\alpha - \left(\alpha\frac{\partial \alpha}{\partial s} + \beta\frac{\partial \beta}{\partial s}\right) = -\left(\frac{k+1}{t} + 2\frac{\partial \alpha}{\partial s} - \beta\kappa\right)\alpha - \beta\frac{\partial \beta}{\partial s}$$

$$\frac{\partial^2 C}{\partial t^2} \cdot N = -\left(\frac{k+1}{t} + \frac{\partial \alpha}{\partial s} - \beta\kappa\right)\beta - \frac{\alpha^2+\beta^2}{2}\kappa + \frac{\lambda k^2 t^{k-2} f}{\rho}$$

Now inserting these acceleration components into (Equation 3.118) yields:

$$\frac{\partial \alpha}{\partial t} = \underbrace{-\left(\frac{k+1}{t} + 2\frac{\partial \alpha}{\partial s} - \beta\kappa\right)\alpha - \beta\frac{\partial \beta}{\partial s}}_{\frac{\partial^2 C}{\partial t^2}\cdot T} + \beta\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right) = \left(-\frac{k+1}{t} - 2\frac{\partial \alpha}{\partial s} + 2\beta\kappa\right)\alpha$$

$$\frac{\partial \beta}{\partial t} = \underbrace{-\left(\frac{k+1}{t} + \frac{\partial \alpha}{\partial s} - \beta\kappa\right)\beta - \frac{\alpha^2+\beta^2}{2}\kappa + \frac{\lambda k^2 t^{k-2} f}{\rho}}_{\frac{\partial^2 C}{\partial t^2}\cdot N} - \alpha\left(\frac{\partial \beta}{\partial s} + \alpha\kappa\right) = -\frac{k+1}{t}\beta - \frac{\partial}{\partial s}(\alpha\beta) + \left(\frac{1}{2}\beta^2 - \frac{3}{2}\alpha^2\right)\kappa + \frac{\lambda k^2 t^{k-2} f}{\rho}$$

Given zero initial velocity (α=0 and β=0), simple inspection shows that α remains zero, leading to the simplified evolution of Equation 3.121.

CALCULATION OF EQUATION EQUATION 3.122

Assuming we represent the evolving curve $C(p,t)$ as the zero level set of an evolving function $\psi(x,t)$ and letting $\hat{\beta}(x,t)$ denote an evolving spatial extension of the evolving normal speed function $\beta(p,t)$ along curve, then we have

$$\psi\left(C(p,t),t\right) = 0 \quad \text{and} \quad \hat{\beta}\left(C(p,t),t\right) = \beta(p,t)$$

Differentiating with respect to t yields:

$$\frac{\partial \psi}{\partial t} + \nabla \psi \cdot \frac{\partial C}{\partial t} = 0 \quad \text{and} \quad \frac{\partial \hat{\beta}}{\partial t} + \nabla \hat{\beta} \cdot \frac{\partial C}{\partial t} = \frac{\partial \beta}{\partial t}$$

Extending the contour evolution $\frac{\partial C}{\partial t} = \beta N$ to other level sets as $\hat{\beta}\hat{N}$, where $\hat{N} = -\frac{\nabla \psi}{\|\nabla \psi\|}$ (noting that this convention for the extension of the inward unit normal requires that the level set function be negative inside the contour and positive outside), yields:

$$\frac{\partial \psi}{\partial t} = \hat{\beta}\|\nabla \psi\| \quad \text{and} \quad \frac{\partial \hat{\beta}}{\partial t} = \frac{\partial \beta}{\partial t} + \nabla \hat{\beta} \cdot \frac{\hat{\beta}\nabla \psi}{\|\nabla \psi\|}$$

which, after substitution of $\frac{\partial \beta}{\partial t}$ using Equation 3.121 results in the level set version of the system in Equation 3.122.

REFERENCES

[1] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

[2] O. Faugeras and O. A. FAUGERAS, *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.

[3] L. Bar, T. Chan, G. Chung, M. Jung, N. Kiryati, R. Mohieddine, N. Sochen, and L. Vese, "Mumford and shah model and its applications to image segmentation and image restoration", Jan. 2014.

[4] S. V. e. Fomin *et al.*, "Calculus of variations", 1963.

[5] D. B. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems", *Communications on pure and applied mathematics*, 1989.

[6] T. F. Chan, J. Shen, and L. Vese, "Variational pde models in image processing", *Notices AMS*, vol. 50, no. 1, pp. 14–26, 2003.

[7] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging", *Journal of mathematical imaging and vision*, vol. 40, no. 1, pp. 120–145, 2011.

[8] T. Goldstein, X. Bresson, and S. Osher, "Geometric applications of the split bregman method: Segmentation and surface reconstruction", *Journal of Scientific Computing*, vol. 45, no. 1-3, pp. 272–293, 2010.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", in *Foundations and Trends in Machine Learning*, vol. 3, 2011, pp. 1–121.

[10] A. J. Yezzi and G. Sundaramoorthi, "Accelerated optimization in the PDE framework: Formulations for the active contour case", *CoRR*, vol. abs/1711.09867, 2017. arXiv: 1711.09867.

[11] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms", *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[12] S. Baker, S. Roth, D. Scharstein, M. J. Black, J. P. Lewis, and R. Szeliski, "A database and evaluation methodology for optical flow", in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[13] A. Yezzi and S. Soatto, "Stereoscopic segmentation", *International Journal of Computer Vision*, vol. 53, no. 1, pp. 31–43, Jun. 2003.

[14] M. Benyamin, J. Calder, G. Sundaramoorthi, and A. J. Yezzi, "Accelerated variational pdes for efficient solution of regularized inversion problems", *J. Math. Imaging Vis.*, vol. 62, no. 1, pp. 10–36, 2020.

[15] A. Yezzi, G. Sundaramoorthi, and M. Benyamin, "Pde acceleration for active contours", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[16] G. S. Anthony J. Yezzi and M. Benyamin, "Accelerated optimization in the PDE framework: Formulations for the active contour case (accepted for publication in 2020)", *SIAM*, vol. abs/1711.09867, 2020.

[17] G. Sundaramoorthi and A. Yezzi, *Accelerated optimization in the pde framework: Formulations for the manifold of diffeomorphisms*, 2018. arXiv: 1804.02307 [math.OC].

[18] J. L. Troutman, *Variational Calclus and Optimal Control.* Springer-Verlag, New York, 1996.

[19] S. Ghadimi and G. Lan, "Accelerated gradient methods for nonconvex nonlinear and stochastic programming", *Math. Program.*, vol. 156, no. 1-2, pp. 59–99, 2016.

[20] W. Krichene, A. Bayen, and P. L. Bartlett, "Accelerated mirror descent in continuous and discrete time", in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., 2015, pp. 2845–2853.

[21] H. Li and Z. Lin, "Accelerated proximal gradient methods for nonconvex programming", in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., 2015, pp. 379–387.

[22] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, 1st ed. Springer Publishing Company, Incorporated, 2014, ISBN: 1461346916, 9781461346913.

[23] N. Flammarion and F. Bach, “From averaging to acceleration, there is only a step-size”, in *Proceedings of Machine Learning Research*, vol. 40, 2015, pp. 658–695.

[24] S. Ruder, “An overview of gradient descent optimization algorithms”, *CoRR*, vol. abs/1609.04747, 2016. arXiv: 1609.04747.

[25] B. T. Polyak, “Some methods of speeding up the convergence of iteration methods”, *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[26] H. Attouch, X. Goudou, and P. Redont, “The heavy ball with friction method, I. The continuous dynamical system: global exploration of the local minima of a real-valued function by asymptotic analysis of a dissipative dynamical system”, *Communications in Contemporary Mathematics*, vol. 2, no. 01, pp. 1–34, 2000.

[27] X. Goudou and J. Munier, “The gradient and heavy ball with friction dynamical systems: The quasiconvex case”, *Mathematical Programming*, vol. 116, no. 1-2, pp. 173–191, 2009.

[28] Y. Nesterov, “A method of solving a convex programming problem with convergence rate o (1/k2)”, in *Soviet Mathematics Doklady*, vol. 27, 1983, pp. 372–376.

[29] N. Yurii, “Primal-dual subgradient methods for convex problems”, *Mathematical programming*, vol. 120, no. 1, pp. 221–259, 2009.

[30] W. Su, S. Boyd, and E. J. Candes, “A differential equation for modeling nesterov’s accelerated gradient method: Theory and insights”, *arXiv preprint arXiv:1503.01243*, 2015.

[31] C. Ward, N. Whitaker, I. Kevrekidis, and P. Kevrekidis, “A toolkit for steady states of nonlinear wave equations: Continuous time nesterov and exponential time differencing schemes”, *arXiv preprint arXiv:1710.05047*, 2017.

[32] A. Wibisono, A. C. Wilson, and M. I. Jordan, “A variational perspective on accelerated methods in optimization”, *Proceedings of the National Academy of Sciences*, p. 201 614 734, 2016.

[33] G. Baravdish, O. Svensson, M. Gulliksson, and Y. Zhang, “Damped second order flow applied to image denoising”, *IMA Journal of Applied Mathematics*, vol. 84, no. 6, pp. 1082–1111, 2019.

[34] J. Calder and A. Yezzi, “Pde acceleration: A convergence rate analysis and applications to obstacle problems”, *Res Math Sci*, pp. 6–35, 2019.

[35] Y. Nesterov, “Gradient methods for minimizing composite functions”, *Math. Program.*, vol. 140, no. 1, pp. 125–161, 2013.

[36] ——, “Accelerating the cubic regularization of newton’s method on convex problems”, *Math. Program.*, vol. 112, no. 1, pp. 159–181, 2008.

[37] Y. Nesterov and B. T. Polyak, “Cubic regularization of newton method and its global performance”, *Math. Program.*, vol. 108, no. 1, pp. 177–205, 2006.

[38] Y. Nesterov, “Smooth minimization of non-smooth functions”, *Math. Program.*, vol. 103, no. 1, pp. 127–152, 2005.

[39] H. Goldstein, C. Poole, and J. Safko, *Classical mechanics*, 2002.

[40] W. Su, S. Boyd, and E. J. Candes, “A differential equation for modeling nesterov’s accelerated gradient method: Theory and insights”, *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 5312–5354, 2016.

[41] R. Szeliski, *Computer vision: algorithms and applications.* Springer Science & Business Media, 2010.

[42] S. Alliney, “A property of the minimum vectors of a regularizing functional defined by means of the absolute norm”, *IEEE transactions on signal processing*, vol. 45, no. 4, pp. 913–917, 1997.

[43] T. F. Chan and S. Esedoglu, “Aspects of total variation regularized l 1 function approximation”, *SIAM Journal on Applied Mathematics*, vol. 65, no. 5, pp. 1817–1837, 2005.

[44] G. Gilboa, N. Sochen, and Y. Y. Zeevi, “Variational denoising of partly textured images by spatially varying constraints”, *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2281–2289, 2006.

[45] T. Le, R. Chartrand, and T. J. Asaki, “A variational approach to reconstructing images corrupted by poisson noise”, *Journal of mathematical imaging and vision*, vol. 27, no. 3, pp. 257–263, 2007.

[46] P. Getreuer, “Rudin-osher-fatemi total variation denoising using split bregman”, *Image Processing On Line*, vol. 2, pp. 74–95, 2012.

[47] Z. Qin, D. Goldfarb, and S. Ma, “An alternating direction method for total variation denoising”, *Optimization Methods and Software*, vol. 30, no. 3, pp. 594–615, 2015.

[48] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks", in *Advances in neural information processing systems*, 2012, pp. 341–349.

[49] R. A. Willoughby, "Solutions of ill-posed problems (a. n. tikhonov and v. y. arsenin)", *SIAM Review*, vol. 21, no. 2, pp. 266–267, 1979. eprint: https://doi.org/10.1137/1021044.

[50] N. Sochen, R. Deriche, and L. Lopez-Perez, "Variational beltrami flows over manifolds", in *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*, IEEE, vol. 1, 2003, pp. I–861.

[51] L. Bar, N. Sochen, and N. Kiryati, "Semi-blind image restoration via mumford-shah regularization", *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 483–493, 2006.

[52] L. Rudin and S. Osher, "Total variation based image restoration with free local constraints", in *Proceedings of 1st International Conference on Image Processing*, vol. 1, Los Alamitos, CA, USA: IEEE Computer Society, Nov. 1994, pp. 31, 32, 33, 34, 35.

[53] M. Tao, J. Yang, and B. He, "Alternating direction algorithms for total variation deconvolution in image reconstruction", *TR0918, Department of Mathematics, Nanjing University*, 2009.

[54] P. Getreuer, "Total variation deconvolution using split bregman", *Image Processing On Line*, vol. 2, pp. 158–174, 2012.

[55] L. Bar, N. Sochen, and N. Kiryati, "Variational pairing of image segmentation and blind restoration", in *European Conference on Computer Vision*, Springer, 2004, pp. 166–177.

[56] D. Mumford and J. Shah, "Boundary detection by minimizing functionals", in *IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, vol. 17, 1985, pp. 137–154.

[57] B. K. Horn and B. G. Schunck, "Determining optical flow", in *Techniques and Applications of Image Understanding*, International Society for Optics and Photonics, vol. 281, 1981, pp. 319–331.

[58] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields", *Computer vision and image understanding*, vol. 63, no. 1, pp. 75–104, 1996.

[59] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping", in *European conference on computer vision*, Springer, 2004, pp. 25–36.

[60] A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers, "An improved algorithm for tv-l 1 optical flow", in *Statistical and geometrical approaches to visual motion analysis*, Springer, 2009, pp. 23–45.

[61] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles", in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, 2010, pp. 2432–2439.

[62] Y. Yang and G. Sundaramoorthi, "Modeling self-occlusions in dynamic shape and appearance tracking", in *Computer Vision (ICCV), 2013 IEEE International Conference on*, IEEE, 2013, pp. 201–208.

[63] Y. Yang, G. Sundaramoorthi, and S. Soatto, "Self-occlusions and disocclusions in causal video object segmentation", in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4408–4416.

[64] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods", *International journal of computer vision*, vol. 61, no. 3, pp. 211–231, 2005.

[65] D. Fortun, P. Bouthemy, and C. Kervrann, "Optical flow modeling and computation: A survey", *Computer Vision and Image Understanding*, vol. 134, pp. 1–21, 2015.

[66] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision", 1981.

[67] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles", in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2432–2439.

[68] P. Anandan, "Measuring visual motion from image sequences", 1987.

[69] D. Geman and G. Reynolds, "Constrained restoration and the recovery of discontinuities", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 3, pp. 367–383, 1992.

[70] A. Blake and A. Zisserman, *Visual reconstruction.* MIT press, 1987.

[71] M. J. Black, "Robust incremental optical flow", Ph.D. dissertation, Verlag nicht ermittelbar, 1992.

[72] M. R. Hestenes, E. Stiefel, *et al.*, *Methods of conjugate gradients for solving linear systems*, 1. NBS Washington, DC, 1952, vol. 49.

[73] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework", *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[74] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite", in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 3354–3361.

[75] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow", *International journal of computer vision*, vol. 92, no. 1, pp. 1–31, 2011.

[76] M. F. Beg, M. I. Miller, A. Trouvé, and L. Younes, "Computing large deformation metric mappings via geodesic flows of diffeomorphisms", *International journal of computer vision*, vol. 61, no. 2, pp. 139–157, 2005.

[77] M. I. Miller, A. Trouvé, and L. Younes, "Geodesic shooting for computational anatomy", *Journal of mathematical imaging and vision*, vol. 24, no. 2, pp. 209–228, 2006.

[78] J. Feydy, B. Charlier, F.-X. Vialard, and G. Peyré, "Optimal transport for diffeomorphic registration", in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2017, pp. 291–299.

[79] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouvé, and G. Peyré, "Interpolating between optimal transport and mmd using sinkhorn divergences", in *The 22nd International Conference on Artificial Intelligence and Statistics*, PMLR, 2019, pp. 2681–2690.

[80] G. Sundaramoorthi, A. Yezzi, and A. Mennucci, "Sobolev active contours", in *Workshop on Variational Geometric and Level Set Methods in Computer Vision*, 2005, pp. 109–120.

[81] G. Charpiat, R. Keriven, J.-P. Pons, and O. Faugeras, "Designing spatially coherent minimizing flows for variational problems based on active contours", in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, IEEE, vol. 2, 2005, pp. 1403–1408.

[82] G. Sundaramoorthi, A. Yezzi, and A. C. Mennucci, "Sobolev active contours", *International Journal of Computer Vision*, vol. 73, no. 3, pp. 345–366, 2007.

[83] G. Charpiat, P. Maurel, J.-P. Pons, R. Keriven, and O. Faugeras, “Generalized gradients: Priors on minimization flows”, *International journal of computer vision*, vol. 73, no. 3, pp. 325–344, 2007.

[84] G. Sundaramoorthi, A. Yezzi, and A. Mennucci, “Coarse-to-fine segmentation and tracking using sobolev active contours”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 851–864, 2008.

[85] G. Sundaramoorthi, A. Yezzi, A. C. Mennucci, and G. Sapiro, “New possibilities with sobolev active contours”, *International journal of computer vision*, vol. 84, no. 2, pp. 113–129, 2009.

[86] A. Mennucci, A. Yezzi, and G. Sundaramoorthi, “Sobolev–type metrics in the space of curves”, *Interfaces and Free Boundaries*, no. 10, pp. 423–445, 2008.

[87] G. Sundaramoorthi, A. Mennucci, S. Soatto, and A. Yezzi, “A new geometric metric in the space of curves, and applications to tracking deforming objects by prediction and filtering”, *SIAM Journal on Imaging Sciences*, vol. 4, no. 1, pp. 109–145, 2011.

[88] Y. Yang and G. Sundaramoorthi, “Shape tracking with occlusions via coarse-to-fine region-based sobolev descent”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 5, pp. 1053–1066, 2015.

[89] E. Klassen, A. Srivastava, M. Mio, and S. H. Joshi, “Analysis of planar shapes using geodesic paths on shape spaces”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 3, pp. 372–383, 2004.

[90] P. W. Michor, D. Mumford, J. Shah, and L. Younes, “A metric on shape space with explicit geodesics”, *arXiv preprint arXiv:0706.4299*, 2007.

[91] M. Micheli, P. W. Michor, and D. Mumford, “Sobolev metrics on diffeomorphism groups and the derived geometry of spaces of submanifolds”, *Izvestiya: Mathematics*, vol. 77, no. 3, p. 541, 2013.

[92] M. Bauer, M. Bruveris, and P. W. Michor, “Overview of the geometries of shape spaces and diffeomorphism groups”, *Journal of Mathematical Imaging and Vision*, vol. 50, no. 1-2, pp. 60–97, 2014.

[93] J. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Material Science*. Cambridge University Press, 1996.

[94] G. Sapiro, *Geometric Partial Differential Equations and Image Analysis*. Cambridge Press, Cambridge, England, 2000.

[95] S. Osher and N. Paragios, *Geometric Level Set Methods in Imaging, Vision and Graphics*. Springer, New York, 2003.

[96] S. Osher and J. Sethian, "Fronts propagation with curvature dependent speed: Algorithms based on hamilton-jacobi formulations", *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[97] T. Chan and L. Vese, "Active contours without edges", *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

[98] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours", *International Journal on Comptuer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[99] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Conformal curvature flows: From phase transistions to active vision", *Archive for Rational Mechanics and Analysis*, vol. 134, pp. 275–301, 1996.

[100] B. K. Horn and B. G. Schunck, "Determining optical flow", *Artifical Intelligence*, vol. 17, pp. 185–203, 1981.

[101] G. Charpiat, R. Keriven, J. Pons, and O. Faugeras, "Designing spatially coherent minimizing flows for variational problems based on active contours", in *Int. Conference Computer Vision*, 2005.

[102] G. Charpiat, O. Faugeras, and R. Keriven, "Approximations of shape metrics and application to shape warping and empirical shape statistics", *Foundations of Computational Mathematics*, vol. 5, no. 1, pp. 1–58, 2005.

[103] G. Sundaramoorthi, A. Yezzi, and A. Mennucci, "Sobolev active contours", *Int. J. Computer Vision*, vol. 7, pp. 345–366, 2007.

[104] G. Sundaramoorthi, J. Jackson, A. Yezzi, and A. Mennucci, "Tracking with sobolev active contours", in *Proc. Computer Vision and Pattern Recognition*, 2006, pp. 674–680.

[105] Y. Yang and G. Sundaramoorthi, "Shape tracking with occlusions via coarse-to-fine region based sobolev descent", *Trans. Pattern Analysis and Machine Intelligence*, 2015.

[106] A. Mennucci, A. Yezzi, and G. Sundaramoorthi, "Properties of sobolev-type metrics in the space of curves", *Interfaces and Free Boundaries*, vol. 10, pp. 423–445, 2008.

[107] G. Sundaramoorthi, A. Mennucci, A. Yezzi, and S. Soatto, "Tracking deforming objects by filtering and prediction in the space of curves", in *Proc. IEEE Conf. Decision and Control*, 2009, pp. 2395–2401.

[108] G. Sundaramoorthi, A. Mennucci, S. Soatto, and A. Yezzi, "A new geometric metric in the space of curves, and applications to tracking deforming objects by prediction and filtering", *SIAM J. Imaging Sciences*, vol. 4, pp. 109–145, 2011.

[109] E. Bardelli, M. Colombo, A. Mennucci, and A. Yezzi, "Multiple object tracking via prediction and filtering with a sobolev-type metric on curves", in *European Conf. Computer Vision*, 2012, pp. 143–152.

[110] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms", in *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR 06)*, IEEE, vol. 1, 2006, pp. 519–528.

[111] B. G. Baumgart, "Geometric modeling for computer vision", STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, Tech. Rep., 1974.

[112] A. Laurentini, "The visual hull concept for silhouette-based image understanding", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 2, pp. 150–162, 1994.

[113] O. Faugeras and R. Keriven, "Variational principles, surface evolution, pdes, level set methods, and the stereo problem", *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 336–344, 1998.

[114] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving", *International journal of computer vision*, vol. 38, no. 3, pp. 199–218, 2000.

[115] D. Cremers, T. Pock, K. Kolev, and A. Chambolle, "Convex relaxation techniques for segmentation, stereo and multiview reconstruction", *Markov Random Fields for Vision and Image Processing*, 2011.

[116] K. Kolev, T. Brox, and D. Cremers, "Fast joint estimation of silhouettes and dense 3d geometry from multiple images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 493–505, 2012.

[117] G. Kuschk and D. Cremers, "Fast and accurate large-scale stereo reconstruction using variational methods", in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 700–707.

[118] G. Gallego, A. Yezzi, F. Fedele, and A. Benetazzo, “A variational stereo method for the three-dimensional reconstruction of ocean waves”, *IEEE transactions on geoscience and remote sensing*, vol. 49, no. 11, pp. 4445–4457, 2011.

[119] Y. Quéau, T. Wu, F. Lauze, J.-D. Durou, and D. Cremers, “A non-convex variational approach to photometric stereo under inaccurate lighting”, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 99–108.

[120] Y. Nesterov *et al.*, *Lectures on convex optimization*. Springer, 2018, vol. 137.

[121] T. Goldstein and S. Osher, “The split bregman method for l1-regularized problems”, *SIAM journal on imaging sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[122] M. Benyamin, J. Calder, G. Sundaramoorthi, and A. J. Yezzi, “Accelerated pde’s for efficient solution of regularized inversion problems”, *arXiv*, vol. 1810.00410, 2018. arXiv: 1810.00410.

[123] N. Sochen, R. Kimmel, and R. Malladi, “A general framework for low level vision”, *IEEE transactions on image processing*, vol. 7, no. 3, pp. 310–318, 1998.

[124] D. Zosso and A. Bustin, “A primal-dual projected gradient algorithm for efficient beltrami regularization”, *Computer Vision and Image Understanding*, pp. 14–52, 2014.

[125] J. Bian, J. H. Siewerdsen, X. Han, E. Y. Sidky, J. L. Prince, C. A. Pelizzari, and X. Pan, “Evaluation of sparse-view reconstruction from flat-panel-detector cone-beam ct”, *Physics in Medicine & Biology*, vol. 55, no. 22, p. 6575, 2010.

[126] P. Combettes and V. Wajs, “Signal recovery by proximal forward-backward splitting”, *Multiscale Modeling and Simulation*, vol. 4, Jan. 2005.

[127] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations”, *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.

[128] M. Niethammer and A. Tannenbaum, “Dynamic geodesic snakes for visual tracking”, in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, IEEE, vol. 1, 2004, pp. I–I.

[129] A. J. Yezzi and S. Soatto, “Stereoscopic segmentation”, in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1, 2001, 59–66 vol.1.

[130] S. R. Fulton, P. E. Ciesielski, and W. H. Schubert, “Multigrid methods for elliptic problems: A review”, *Monthly Weather Review*, vol. 114, no. 5, pp. 943–959, 1986.