Universität des Saarlandes

Bachelor Thesis

Exploring Corner Regions as Inpainting Domain for PDE-based Image Compression

---

Author:
Daniel Gusenburger

Supervisor:
Prof. Joachim Weickert

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Mathematical Image Analysis Group
Department of Computer Science

April 3, 2020

UNIVERSITÄT DES SAARLANDES

Abstract

Faculty of Mathematics and Computer Science
Department of Computer Science

Bachelor of Science

Exploring Corner Regions as Inpainting Domain for PDE-based Image Compression

by Daniel Gusenburger

Contents

Chapter 1

Introduction

Before we dive in, let me first explain what this topic is about and try to motivate the thought process behind it as well shortly explain what you are going to read about on the next pages.

1.1 Motivation

As technology evolves, the quality and resolution of digital images improve as well. But as the quality increases so does the memory required to store the image on a hard drive. To counteract this increase in disk space usage, people have tried to reduce the sizes of digital images a lot in the last decades.
One of the most successful and probably most well known *codecs* is **JPEG** and its successor **JPEG 2000**. Both are lossy image compression methods known for fairly high compression rates while still providing a reasonably image quality. For higher compression rates however, the quality deteriorates pretty quickly and the infamous "block artifacts" are being introduced. As a remedy, a new method for image compression has been developed in the last years that aims to create better looking images for higher compression rates than JPEG and even JPEG2000.
This new method roughly works by selecting a small amount of pixels to keep and then filling in the gaps in the reconstruction/decompression step.
As one can imagine, selecting the right data is a fairly minute process and one has to carefully select the pixels to keep. Even though there has been a lot of work done in this area, the selection can still be improved.
In the past, the usefulness of corners for this process was proven in [1] even though the method proposed in this work would not surpass JPEG's abilities. Nonetheless, we want to build on it and explore how keeping larger regions of data around corners plays out in this process.

1.2 Related Work

Next up, we are going to discuss some work related to this thesis to see what advances have been done in the last years and what our work is built upon.

1.2.1 PDE-based Image Compression

In 2005, Galić et al. first introduced an alternative image compression method using PDE-based inpainting as a serious alternative to more classical approaches like JPEG and JPEG 2000 [2]. In this work, the authors showed the inpainting capabilities of nonlinear anisotropic diffusion, specifically of a diffusion process called *edge-enhancing diffusion*, or short EED. The specifics of this process will be covered in 2.3.3.
For data selection they used an *adaptive sparsification scheme relying on B-tree triangular coding (BTTC)*, hence the name *BTTC-EED* [2], as an easy to implement and fast compression method [3]. With this fairly simple approach they were already able to outperform JPEG visually for high compression rates and comic-style images [2].
Improving on this, the authors published a new paper in 2008, adding a number of additional procedures to the compression phase, with which they were finally able to come close to the quality of JPEG 2000 [4]. Finally, in 2009, Schmaltz et al. optimised the ideas even further, building the so called **R-EED** codec with which they could even beat JPEG 2000 [5]. The main differences between [2] and [5] are the addition of several procedures to optimise the data set that is kept for inpainting in the decompression step. To roughly summarise the whole compression phase [5]:
First, an initial set of points is gathered by using a rectangular subdivision (instead of the previous triangular subdivision) of the image. This works by recursively splitting the image in half whenever the reconstruction using only the boundary points exceeds a certain error threshold. The reconstruction is also done using EED inpainting. After obtaining the initial data set, the brightness values of each of the kept pixels is rescaled to $[0, 255]$ to eliminate possible quantisation artifacts. Due to this brightness rescaling, the optimal contrast parameter for the decompression phase may change and thus has to be adjusted as well. This is generally done alternating between optimising points and the contrast parameter until a certain convergence criterion is met. As a last step, the authors invert the inpainting mask and perform the inpainting process to fill in the kept data as a means to increase the coherence between the optimised pixels and the original image.

All of these measures serve the purpose of decreasing the *mean squared error (MSE)* to a level where the proposed codec is able to outperform JPEG 2000 for compression rates higher than **43** : **1**.

1.2.2 Inpainting

Inpainting as a technique is nothing new, it existed since a long time in the form of e.g. restoration of old images and film. In 2000, Bertalmio et al first proposed an algorithm to digitally inpaint images without user intervention. After consulting actual experts in image restoration they came up with a method imitating human restorators.
The main idea behind their method is to continue the structure surrounding the gap into it and simultaneously fill in the different regions in each gap with the colour at its boundaries[6]. Although it produces good looking images without obvious artifacts, it lacks the ability to reproduce texture. Furthermore, they proposed to use second order PDEs instead of the high order PDEs they used to solve the inpainting problem. Galić et al touched on this issue in 2008, proposing to use EED because of its inpainting capabilities as a replacement for the higher order PDE[4]. However, they only used EED for inpainting instead of interleaving a PDE based inpainting approach with a mean curvature motion model as proposed in [6]. This was featured in another work that I will cover in the next section.

1.2.3 Image features in Inpainting

Semantically, edges and corners are the most important features of an image. Because of this, there are multiple publications trying to exploit the semantic importance of these features for image inpainting. For example in 2010, Mainberger et al published a paper on the reconstruction of images using only relevant edges and homogeneous diffusion. Building on their work from 2009, the authors were able to successfully reconstruct cartoonish images from only a set of edges they detectedj using the Marr-Hildreth edge detector[7]. In contrast to other inpainting methods that rely on sparse images as their inpainting domain and therefore need to use more sophisticated PDEs in order to successfully reconstruct the image, the algorithm described in this work is built around a simple homogeneous diffusion equation[8]. Their reasoning behind this is that homogeneous diffusion is *one of the analytically best understood inpainting approaches*([8]) as well as, because of its simplicity, computationally the least challenging out of all PDE-based approaches.

Another approach by Zimmer that I already mentioned in 1.1 proved the importance of corners for image inpainting in image compression[1]. Even though they could not beat the quality of JPEG, it still serves as a valuable foundation for future work. Their approach was to create a sparse image from a set of what they called *corner regions* which essentially is the set of pixels directly neighbouring a cornerd detected by the Förstner-Harris corner detector[9]. For the inpainting in the decompression phase, they came up with a more complex version of EED-based inpainting by interleaving it with *Mean Curvature Motion (MCM)* which, in the past, has proven itself valuable especially for inpainting larger regions [6].

1.3 Organisation

The thesis is organised as follows:
First, I will introduce some mathematical concepts such as the structure tensor and diffusion processes in 2 as well as talk about the general theory behind this topic. Afterwards in 3, we will discuss discretisation strategies and how the parameters for corner detection and inpainting were chosen. In 4, I will shortly go over the testing framework I implemented to more efficiently generate test images and simultaneously test the procedure on these images and then show some examples. Last, but not least, we will discuss the shortcomings and future work in 5.

Chapter 2

Theory

2.1 Basics

The concepts used in this thesis require some prior knowledge about basic calculus and linear algebra as well as some more advanced topics that will be introduced in the following sections. But before introducing corner detection and diffusion, we have to first define what an image is mathematically.
A *grey value image* is defined as a function $f : \Omega \to [0, 255]$ where $\Omega \subset \mathbb{R}^2$ is a rectangular subset of $\mathbb{R}^2$ of size $n_x \times n_y$, wheras a *colour image* is defined as a vector-valued function $f : \Omega \to [0, 255]^3$. For the sake of simplicity, we will focus on grey value images as most of the results can easily be transferred to vector-valued images.

2.1.1 Image gradient

One of the most important operations on functions in Image Processing is *partial differentiation*. The partial derivative of an image $f : \Omega \to [0, 255]$ in x-direction is herein denoted as f_x or synonymously as $\partial_x f$ and defined as

$$f_x(x, y) = \partial_x f(x, y) = \frac{\partial f}{\partial x}(x, y) := \lim_{h \to 0} \frac{f(x + h, y) - f(x, y)}{h} \tag{2.1}$$

The *gradient* of an image f is the vector containing both partial image derivatives. In multivariable calculus, the gradient of a function is an important tool to find the (both local and global) extrema of a function similar to the first derivative for a function with a single variable.

$$\mathbf{grad}(f) = \boldsymbol{\nabla} f := (f_x, f_y)^\top \tag{2.2}$$

The gradient always points in the direction of the steepest ascend/descend, it is the tangent vector to the surface at the given location[10]. Note that the gradient of a function is a vector-valued function and not a vector.

2.1.2 Convolution

Another operation from calculus that we will need is the *convolution operator.*

$$(f * g)(x) := \int_{\mathbb{R}} f(x - x')g(x')dx' \tag{2.3}$$

$$(f * g)(x, y) := \iint_{\mathbb{R}} f(x - x', y - y')g(x', y')dx'dy' \tag{2.4}$$

Convolution is especially useful in Image and Signal Processing to design so called linear filters such as a moving average or smoothing operation[11][12]. As a matter of fact, in a later section we will need the convolution as a tool to smooth our image to reduce noise artifacts. To achieve this, we will use a *Gaussian convolution*, i.e. a convolution with a *Gaussian kernel* which is basically just a two-dimensional Gaussian function with a certain standard deviation[11]:

$$K_\sigma(x, y) := \frac{1}{2\pi\sigma^2} \exp\left(\frac{-(x-\mu_1)^2 - (y-\mu_2)^2}{2\sigma^2}\right) \tag{2.5}$$

In the definition above, μ_1, μ_2 are the mean values in each direction. For the rest of this thesis, an image f convolved with a Gaussian with standard deviation σ will be denoted by

$$f_\sigma := K_\sigma * f$$

Note that because of the symmetry of the convolution, it would have been perfectly fine to write it as $f * K_\sigma$. If the reader wants to know more about convolution and its properties, they are kindly referred to [12].

2.2 The Structure Tensor

For some applications only the gradient of an image does not give us enough information. The gradient on its own is mostly just used as an edge detector, hence we need to come up with something else for e.g. corner detection[13]. One option is the so called *structure tensor*, a matrix that contains information about the surrounding region at a specific position. With the structure tensor, or rather its eigenvalues (cf. 2.2.2), one is able to distinguish between flat regions, edges and corners.

2.2.1 Definition

!! Reconsider the definition, maybe rewrite this paragraph later !!

The structure tensor is defined as a matrix whose eigenvectors tell us the direction of both the largest and smallest grey value change. Mathematically, we can model this as an optimisation problem:

Let u be a grey value image. We want to find a unit vector $\mathbf{n} \in \mathbb{R}^2$ that is 'most parallel' or 'most orthogonal' to the gradient $\boldsymbol{\nabla} u$ within a circle of radius $\rho > 0$, i.e. one wants to optimise the function

$$E(\mathbf{n}) = \int\limits_{B_\rho(x,y)} \left(\mathbf{n}^\top \boldsymbol{\nabla} u\right)^2 dx'dy' \tag{2.6}$$

$$= \mathbf{n}^\top \left(\int\limits_{B_\rho(x,y)} \boldsymbol{\nabla} u \boldsymbol{\nabla} u^\top dx'dy' \right) \mathbf{n} \tag{2.7}$$

This function is also called the *local autocorrelation function/local average contrast*[9], [13]. Since (2.7) is a quadratic form of the matrix

$$M_\rho(\boldsymbol{\nabla} u) := \int\limits_{B_\rho(x,y)} \boldsymbol{\nabla} u \boldsymbol{\nabla} u^\top dx'dy$$

such an optimal unit vector is by definition also the eigenvector to the smallest/largest eigenvalue of $M_\rho(\boldsymbol{\nabla} u)$[13]. The matrix $M_\rho(\boldsymbol{\nabla} u)$ can also be seen as a component-wise convolution with the indicator function

$$b_\rho(x,y) = \begin{cases} 1 & x^2 + y^2 \leq \rho^2 \\ 0 & \text{else} \end{cases}$$

However, as the author stated in [9], using this *binary window function* leads to a noisy response and they therefore suggest using a *Gaussian window function* with standard deviation ρ. This parameter is also called the *integration scale* and determines how localised the structure information is[13]. This ultimately leads to the definition

$$\mathbf{J}_\rho(\boldsymbol{\nabla} u) := K_\rho * \left(\boldsymbol{\nabla} u \boldsymbol{\nabla} u^\top\right) \tag{2.8}$$

To keep things simpler, I will omit the brackets and just simply use $\mathbf{J}_\rho$ as the structure tensor.

2.2.2 Usage in Corner Detection

The structure tensor is a symmetric matrix and thus possesses orthonormal eigenvectors $\mathbf{v}_1, \mathbf{v}_2$ with real-valued eigenvalues $\lambda_1, \lambda_2 \geq 0$. [13] As mentioned in the preface to this section, we can use these eigenvalues to distinguish between corners, edges and flat regions as seen in figure 2.1. In total, we have to deal with 3 different cases:

1. λ_1, λ_2 are both small

2. one of the eigenvalues is significantly larger then the other one

3. both eigenvalues are significantly larger than 0

!! Better explanation !!

In the first case, the region inside the Gaussian window is flat, in the second case there is a prominent edge and in the last one, there is a corner. There are

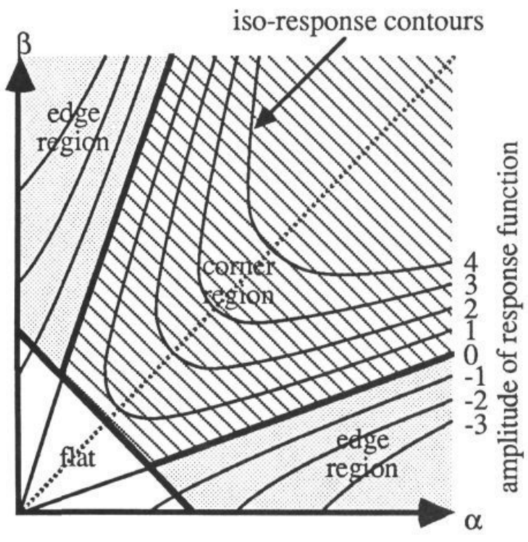

FIGURE 2.1: Visualisation of distinction of image features using the eigenvalues of the structure tensor. α, β are equivalent to the eigenvalues λ_1, λ_2. Source: [9]

different approaches to finding out whether the current location is a corner or an edge. The simplest looking approach simply thresholds the smaller eigenvalue against some artificial threshold. The set of local maxima is then the set of corners for the image[14].

!! Find sources for Rohr and Förstner !!

However, this approach requires to compute both eigenvalues and can thus be fairly expensive.

A cheaper approach would be to either threshold the trace

$$tr(\mathbf{J}_\rho) := j_{1,1} + j_{2,2} = \lambda_1 + \lambda_2$$

as proposed by Rohr, 1987 or the determinant

$$det(\mathbf{J}_\rho) := j_{1,1}j_{2,2} - j_{1,2}^2 = \lambda_1\lambda_2$$

as proposed by Harris and Förstner, 1988 and 1986 respectively[9]. Both of these approaches do not need to explicitly compute the eigenvalues of the structure tensor and are thus not as computationally invested. The differences between the approaches is that the first one requires the trace by itself to be a local maximum whereas in the second approach, $\frac{det(\mathbf{J}_\rho)}{tr(\mathbf{J}_\rho)}$ needs to be a local maximum.

For the detection of relevant corners in the data selection phase, I mainly used the approach of Förstner/Harris as well as the approach of Rohr even though the Tomasi-Kanade approach was an option and has also been tested as we will see later in chapter 4. However, it has not proven as successful as the other two methods during the initial testing phase.

2.3 Diffusion

2.3.1 What is Diffusion?

2.3.2 Linear Diffusion

2.3.3 Nonlinear Diffusion

2.4 Basics of Inpainting

2.4.1 EED-based Inpainting

Chapter 3

Implementation

3.1 Discretisation

3.1.1 Discrete Images

3.1.2 Numerical Differentiation

3.1.3 Numerical Schemes for Diffusion

3.2 Parameter Selection

3.2.1 Corner Detection

3.2.2 Inpainting

3.3 Adaptations

Chapter 4

Results

4.1 Testing framework

4.2 Test images

4.3 Experiments

4.4 Results

Chapter 5

Conclusion and Outlook

5.1 What works well

5.2 What does not

5.3 Future Work

Bibliography

[1] Henning Lars Zimmer. “PDE-based Image Compression using Corner Information”. MA thesis. 2007.

[2] Irena Galić, Joachim Weickert, Martin Welk, et al. *Towards PDE-Based Image Compression*. 2005.

[3] Riccardo Distasi, Michele Nappi, and Sergio Vitulano. *Image Compression by-Tree Triangular Coding*. 1997.

[4] Irena Galić, Joachim Weickert, Martin Welk, et al. *Image Compression with Anisotropic Diffusion*. 2008.

[5] Christian Schmaltz, Joachim Weickert, and Andrés Bruhn. “Beating the quality of JPEG 2000 with anisotropic diffusion”. In: *Pattern Recognition, volume 5748 of Lecture Notes in Computer Science*. Springer, 2009, pp. 452–461.

[6] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, et al. “Image Inpainting”. In: *SIGGRAPH ’00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[7] Ellen C. Hildreth. “Edge Detection”. In: 207 (1985), pp. 187–217.

[8] Markus Mainberger, Andrés Bruhn, Joachim Weickert, et al. *Edge-Based Compression of Cartoon-like Images with Homogeneous Diffusion*. 2010.

[9] Christopher G. Harris and Mike Stephens. “A Combined Corner and Edge Detector”. In: *Alvey Vision Conference*. 1988, pp. 147–151.

[10] Joachim Weickert. *Mathematik für Informatiker III*. Lecture Notes. 2016. URL: https://www.mia.uni-saarland.de/Teaching/MFI16/MfI3_Skript.pdf.

[11] Joachim Weickert. “Foundations II: Degradations in Digital Images”. In: *Image Processing and Computer Vision*. Lecture note. Saarland University, 2019. URL: https://www.mia.uni-saarland.de/Teaching/IPCV19/ipcv19-02.pdf.

[12] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. 1997. URL: http://www.dspguide.com/pdfbook.htm.

[13] Joachim Weickert. "Linear Filters III: Detection of Edges and Corners". In: *Image Processing and Computer Vision*. Lecture note. Saarland University, 2019. URL: https://www.mia.uni-saarland.de/Teaching/IPCV19/ipcv19-13.pdf.

[14] Jianbo Shi and Carlo Tomasi. "Good Features to Track". In: 1994, pp. 593–600.