Datalogisk Institut
KØBENHAVNS UNIVERSITET

DIKU

Video Upscaling
Using Variational Methods

Sune Høgild Keller

Technical Report no. 07-03
ISSN: 0107-8283

Dept. of Computer Science
University of Copenhagen • Universitetsparken 1
DK-2100 Copenhagen • Denmark

Video Upscaling Using Variational Methods

Sune Høgild Keller

The Image Group, Department of Computer Science
Faculty of Science, University of Copenhagen

2007

Preface

This constitutes Sune Høgild Keller's PhD thesis. The thesis is submitted at the Department of Computer Science (DIKU), Faculty of Science, University of Copenhagen in partial fulfillment of the requirements for the degree of doctor of philosophy.

This thesis was successfully defended on December 3, 2007. The evaluation committee was: Professor Gerard de Haan (Technical University Eindhoven and Philips Research), PhD Senior Researcher David Tschumperlé (GREYC Lab, Caen, France) and Associate Professor Jon Sporring (DIKU, University of Copenhagen).

The first part of this work has been carried out at The Image Group, IT University of Copenhagen (ITU) in the period September 2004 to December 2006. The remaining work has been done at the Image Group, Department of Computer Science, Faculty of Science, University of Copenhagen in the period January 2007 to September 2007. All work has been supervised by Professor Mads Nielsen who is currently affiliated with the same group as me and director of Nordic Bioscience Imaging (NBI) a division of Nordic Bioscience A/S, Herlev, Denmark. Co-supervision has been done by Senior Researcher François Bernhard Lauze of NBI (until December 2006 Assistant Professor with the Image Group at ITU).

It was the parallel between holes in interlaced video and holes in old films that gave birth to this PhD project. I was annoyed by the pour image quality on a standard interlaced tv-sets and Mads Nielsen and François Lauze were looking for new problems to tackle with variational methods. Thus the idea for the master thesis of Sune Høgild Keller [57] was borne. Then followed a patent application (US [66] and PCT [67]) on the idea of using energy minimization and variational methods for video upscaling in general and a grant making this PhD thesis possible.

Author Contact Information

Telephone: +45 26 79 09 79
Email: sunebio@itu.dk
Home page: image.diku.dk/sunebio

Acknowledgements

So many people have helped me get to where I am today, but only a few have been mentioned here. I have a wonderful family and great friends, to whom I count almost all my colleagues: That's how it goes when one has been lucky enough to work in both the former Image Group at ITU and in the Image Group at DIKU. To all of you, mentioned here or not, thanks!

Mads Nielsen provided me with the tools of how to fight the annoyance bad viewing experiences in my home cinema – sunebio. You were a great coach both professionally and mentally: You reach for the stars and I try to tag along. Thanks for a great skiing trip to Colorado!

François Lauze, you are French with a big fat **F** and your trademark 'double-wrues'. We didn't always see eye to eye on how things should be done, but I believe it strengthened our joint work. You brought the math back into my life in the best of ways and you supervised me way beyond all requirements, mon ami.

Jenny, my ITU office buddy for two years at ITU, thank you for a great friendship and many shared experiences in the life of PhD students, heja Sverige. Thanks also to Rabia who gave me first class company while Jenny was away and to Aditya, David and Lars S. for doing the same at DIKU.

Kim Steenstrup Pedersen, you gave great support when I did it 'my way' scientifically. Grumse, you told me how to keep François and Mads on a short leash. Marco who always listened when I needed to bitch or just talk, I think you are the colleague/friend with whom I shared the most good times – and beers. Thanks for the hill hike in Hofgeismar. Marleen, I really appreciate your friendship and professional advice – and how you made it crystal clear to me how good life as a PhD is in Copenhagen – at least compared to the Netherlands. Eugenio, you tried to charm the ladies old and young but you also charmed this guy into a great friendship. And I could go on about the rest of you: Eva, Anne, Ole, Erik, Lars C.-H., Michael, Jakob, Arjen, Arish, Pieter, Corné, Nanna, Pechin, Lauge and Vladlena. To Peter Jo, the rest of the 'old' DIKU Image Group and Stig, thanks for making me feel welcome to DIKU.

Outside our own little duck pond here in Copenhagen, thanks to Sarang Joshi and Steve Pizer and his wife Lynn for given me a feel of the university life at UNC Chapel Hill. To the Saarbrücken guys, Joachim, Andres and Martin for great discussions in Copenhagen and in scale spaces. And talking about scale spaces, thanks for sharing thoughts on science and PhD life to the Eindhoven gang, Bart, Erik, Bram, Frans and Evgeniya.

Then there are those who do not play with images. Camilla Torp-Smith, Lotte Møller, Gitte Hornstrup Dahl, Camilla Jørgensen and Marianne Henriksen, you all survive daily contact with us crazy scientist, thank you all for great help and advice.

My Family. My mother who has always been there for me, my sometimes annoying kid sister, I love you. My grandfather Ingolf who has taught through setting an example, how to keep my feet firmly planted on the ground, my late grandmother Ellen Marie who always paid me for my grades good or bad, my late uncle Gerwin for making me interested in technology and science and my father who did the same posthumously. Martin, Steffen, Helle, Gerda, Tom, Sofie and the rest of you.

Whenever I went flat on the batteries, I have always been able to go and reload in Krampamåla, paradise on earth in deep forests of Sweden. Thanks for many great times Gitte, Benny, Lasse, Thea, Susanne, Claus, Astrid and Christine.

Finally the biggest thanks of all to Bente, my wonderful fiancée, who has made the vast majority of my days the last seven years a perfect balance between a professionally fulfilling work life and a wonderful private life. Without you this dissertation would not have been, I dedicate it to you – det her er til dig, tak!

København, December 2007.

Abstract

In this thesis we do image sequence upscaling using variational methods. We have developed, implemented and tested the three main elements of the upscaling part of a video processor using variational methods plus a non-variational preprocessing method detecting the scan format of the input video. The three upscalings needed are deinterlacing (DI), which is creating the never recorded every other line in interlaced video, video super resolution (VSR), which is increasing the spatial resolution of each frame in the video throughput, and temporal super resolution (TSR), which changes the frame rate of a sequence by creating fully new frames at the correct temporal positions.

Our variational upscaling methods have been derived from a Bayesian inference framework for image sequence restoration and enhancement. The framework dictates simultaneous computation of the flow and intensities of the repaired and/or enhanced output sequences. The framework was first suggested for image sequence inpainting in [65].

From the framework we derive a motion adaptive (MA) deinterlacer and a motion compensated (MC) deinterlacer and test them together with a selection of known deinterlacers. To illustrate the need for MC deinterlacing the interlacing problem is introduced. It cannot be solved by MA deinterlacers or any simpler deinterlacers but only by MC deinterlacers. The major hurdle in doing MC deinterlacing is reliable optical flow computations on interlaced video. We discuss a number of strategies for computing optical flows on interlaced video hoping to shed some light on this problem. We produce results on real world video data with our variational MC deinterlacer that even in many difficult cases are indistinguishable from the ground truth.

Producing high image quality on high definition (HD) displays when showing standard definition (SD) material is a problem of upscaling frame resolutions from low resolution (LR) to high resolution (HR) and is as such a super resolution (SR) problem, or as we prefer: *Video super resolution* (VSR). In technology available today the problem is typically solved using simple spatial interpolation. Using motion compensated methods instead will allow for information transport along the optical flow trajectories of the video and increase the level of detail and sharpness in the high resolution output. We present a variational motion compensated VSR method derived from our Bayesian framework that simultaneously computes the desired high resolution video and a high resolution flow field to increase accuracy of the temporal information transport. Creating super resolution flows has to our knowledge not been done before. Most advanced SR methods found in literature cannot be applied to general video with arbitrary scene content and/or arbitrary optical flows as it is possible with our simultaneous VSR method, which also allows for arbitrary discrete magnification factors. We show in test that our variational simultaneous VSR algorithm outperforms other SR methods applicable to our general video problem, and we also attempt to break the limits of super resolution [2] in our experiments by increasing the frame resolution eight times in both height and width (8x8 VSR).

Temporal super resolution (TSR), the ability to convert video from one frame rate to another is a key functionality in a modern video processing systems. A different and often higher frame rate than what is recorded is desired for high frame rate displays, for video/film format conversion, or for super slow-motion.

We present a novel motion compensated TSR algorithm using variational methods for both optical flow calculations and the actual new frame interpolation as naturally derived from our Bayesian framework. The flow and intensities are calculated simultaneously in a multiresolution setting. We discuss what output quality is desired from a TSR algorithm. A major problem in watching video on large and bright displays is that the motion of high contrast edges often seem jerky. We test an implementation of our algorithm focussing on getting the motion of high contrast edges to seem smooth and natural by doubling the frame rate, thus reestablishing the effect of motion pictures.

We introduce an algorithm – commonly known as a film mode detector – for separating progressive source video from interlaced source video. Due to interlacing artifacts in the presence of motion, a difference in isophote curvature can be measured and a threshold for effective classification can be set. This can be used in a video upscaling to ensure high quality output as one can determine if deinterlacing is necessary or not.

Finally we discus the remaining unsolved problems in variational upscaling (hopefully) subject to future development. The most essential is realtime implementations of our algorithms, which from our assessment are doable with today's technology.

List of Publications

Peer-Reviewed Conference Papers

- Sune Høgild Keller, François Lauze and Mads Nielsen, *Motion Compensated Video Super Resolution*, Scale Space and Variational Methods in Computer Vision, SSVM 2007 Proceedings, F. Sgallari, A. Murli and N. Paragios, editors, LNCS 4485, pages 801-812, Berlin, Springer, 2007.
- Sune Høgild Keller, François Lauze and Mads Nielsen, *Variational Motion Compensated Deinterlacing*, 3rd Workshop on Statistical Methods in Multi-Image and Video Processing (SMVP), 9th ECCV Workshop Proceedings, 2006.
- Sune Høgild Keller, Kim S. Pedersen and François Lauze, *Detecting Interlaced or Progressive Source of Video*, Proceedings of the 2005 IEEE Seventh Workshop on Multimedia Signal Processing, 2005.
- Sune Høgild Keller, François Lauze and Mads Nielsen, *A Total Variation Motion Adaptive Deinterlacing Scheme*, Scale Space and PDE Methods in Computer Vision, 5th International Conference, Scale-Space 2005, Proceedings, R. Kimmel, N. Sochen and J. Weickert, editors, LNCS 3459, pages 408-419, Berlin, Springer, 2005.

Patents

- François Lauze, Sune Høgild Keller and Mads Nielsen, *A Method of Adding Information to a Frame or a Field in a Video Sequence*, PCT Patent Application WO-A-2006/035072 (PCT/EP2005/054957), Sept. 30, 2005.
- François Lauze, Sune Høgild Keller and Mads Nielsen, *A Method of Adding Information to a Frame or a Field in a Video Sequence*, U.S. Patent Application US 11/239697, Sept. 30, 2005.

Technical Report

- Sune Høgild Keller, François Lauze and Mads Nielsen, *Variational Deinterlacing*, IT University Technical Report Series, TR-2006-90, ISBN 87-7949-134-0, 2006.

Contents

Chapter 1

Introduction

1.1 The Challenge

The purpose of this PhD project has been to develop the elements needed for the upscaling part of a video processor using variational methods. The three types of needed upscalings are deinterlacing (DI), which is creating the never recorded lines in interlaced video, also known as interlaced to progressive conversion, video super resolution (VSR), which is increasing the spatial resolution of each frame in the video throughput, and temporal super resolution (TSR), which changes the frame rate of a sequence by creating fully new frames at the temporal positions where they are needed.

De-noising, image reconstruction and inpainting are some of the problems earlier tackled with success using variational methods. From the work done by Lauze in [64] we had the platform of doing variational based inpainting, and to put it simple, the job was no longer to repair pixels but to create new ones. Holes in images, whether it is the result of a damage to a film strip or lines not recorded in an interlaced video signal, are in a general sense the same, just holes. But as this work will show there is a is great deal more to doing upscaling than just applying an inpainting algorithm to the problem. When actually solving the problems it becomes clear that the three types of upscalings, DI, VSR and TSR are three quite different problems, making variational DI, VSR and TSR three challenging problems.

1.2 Motivation

Today's video upscalers do not produce outputs of a quality good enough to please the human observers, and thus improved deinterlacers, frame rate converters (TSR) and spatial upscalers (VSR) are sought for in both broadcasting, film/tv production and home consumer products. Variational methods, although still only being an emerging technology, have already proven their worth in inpainting (see [64]) and thus it seemed obvious to us that variational methods would give high quality results if applied to the problem of video upscaling.

By the time we started our work on deinterlacing (late 2003), this problem had only been attacked from an electronic engineering point of view and variational methods had never been applied to the problem. Along with the work of Tschumperlé and Besserer in [102] (using 2D spatial structure tensors) we were the first to introduce variational approaches to the field of deinterlacing. Bayesian formulations, energy minimizations and total variation was not unknown to the field of super resolution (SR), but our ideas of a motion compensated approach with no limitations on the flow, and an algorithm producing simultaneously a) a high resolution sequence and b) a corresponding high resolution optical flow, are new. And the integration in our variational framework is also new. When it comes to temporal super resolution, motion compensated methods are the only way to get reasonable results, but mostly block matching flow estimation and simple temporal interpolation are used. Thus the simultaneous calculation of both intensities and flows using variational methods is new. All in all the basics of a full research project was there, and work on this PhD thesis was started.

The general idea of variational, energy minimizing video upscaling given in the following chapter is also what is covered by the US and PCT patent applications [66] and [67]. It is our hope to see our algorithms implemented and used in 'real world' video processing systems in the (near) future. This system could be placed in the editing room of a film/tv production company, at a broadcasting company doing HDTV transmissions or in a private home incorporated in video playback devices, displays, high end stand alone video processors like Faroudja's DVP-1010 (see Chapter 3) or other electronic media devices.

Our focus throughout the process of doing this work has been to lift the bar on what can be achieved in terms of output image quality from video upscalers. The most likely reasons why variational methods have never been tried in upscaling before, are that they are considered too computationally expensive and are relatively unknown in the field of electronic engineering. But the fast growth in the performance/price ratio of field-programmable arrays (FPGAs) and other multiprocessor technologies along with the work of Andres Bruhn and others (see e.g. [11]) on realtime variational optical flow computations on standard PCs, have assured us that realtime variational video upscaling is within reach at a reasonable price even today.

1.3 Organization of this Dissertation and How to Read It

The basic idea of this project to build the elements of an upscaling video processor is reflected in the organization of this thesis as the four main chapter each represents an element.

- In Chapter 2, before going into details with each element, we present the background of our work: We look at human vision, basic technical details and the general idea of building an inference machine, the philosophy and method leading to a model of image sequence enhancement and reconstruction in general and a video upscaling solution in particular.
- Chapter 3 is on deinterlacing. An extension of the work done on vari-

ational motion adaptive (MA) deinterlacing in the master thesis [57] of Sune Høgild Keller is presented along with the work on it successor, variational motion compensated (MC) deinterlacing. The chapter is intended for future journal publication in IEEE Transactions on Image Processing. Earlier versions of the work has been published in [55] and [53].

- Chapter 4 is on video super resolution (VSR) and also intended for journal publication. Initial work on variational video super resolution not producing high resolution flows along with the high resolution intensity sequences was published in [54].
- Chapter 5 is on temporal super resolution (TSR). Although intended for journal publication, it is written in integrated chapter form, requiring prior reading of Chapter 2 to get the in-depth picture. Chapter 3 and Chapter 4 can be read independently of any other chapters, but readers will benefit from also reading Chapter 2.
- In Chapter 6 we present a scan format detector separating interlaced video from progressive video by measuring curvature statistics. The scan format detection is a crucial preprocessing step in upscaling, determining whether or not to deinterlace a given input. This work was earlier published in [56].

The ordering of the Chapters 3–5 above suggests our basic idea of how to cascade the three types of upscaling in a video processor. If the input is progressive to begin with but packed in an interlaced broadcast signal as is the case with for instance cinematic movies, one needs to bypass the deinterlacing simply just rejoining the separated frames. Determining whether the signal is progressive or interlaced is not straightforward and that is where the method of Chapter 6 comes in. Although format detection is a preprocessing step, we have placed in the back as it is the smallest of the four contributions represented by the Chapters 3–6, and the chapters 3–5 relates directly to the general idea presented in Chapter 2 and each other. Since Chapters 3, 4 and 6 are written in the form of stand alone publications, they can be read individually. Reading the full thesis some redundance will inevitable occur, mainly in the theoretical parts of the chapters. To the reader less experienced with variational methods this might serve as a help to understand the field.

Finally, in Chapter 7 we summarize and look ahead, discussing how the work presented in this thesis can possibly be used and improved in the future, for instance we look at how to organize and integrate the three upscalings steps deinterlacing, video super resolution and temporal super resolution.

Chapter 2

Background

In this chapter we will go through the most important parts of the foundation of knowledge on which our work stands. We will start by giving an overview of the technical details in video and broadcasting today in Section 2.1, then we look at human vision in Section 2.2 as it is the sense we wish to optimally stimulate with our results. In Section 2.3 we then look at how to measure the quality of our work by it deviation from a ground truth or by how pleasing it is to the human visual system. Section 2.4 is on color versus grey scale video upscaling and Section 2.5 discuss work on variational optical flow, which is integrated into our variational video upscaling framework presented in Section 2.6.

2.1 Technical Overview

Since this section is dense with facts upon facts, to keep it fairly readable, we have not stated references for each and every small piece of information. As main sources of information we have used [4], [70], [82], [97], [98] and [107], and we only give the main, necessary information leaving out the details not needed explicitly in this thesis.

In Europe the dominating broadcasting standards are PAL (Phase Alternating Line) and SECAM (Séquentiel couleur à mémoire) which in their analog forms have an apparent resolution of 576 horizontal lines of video. They both transmit 50 interlaced fields per second, each field containing 288 lines of video, alternatingly the even numbered or the odd numbered horizontal lines (illustration given in Figure 3.1(b) in Chapter 3 on deinterlacing). The name interlacing comes from the fact that the transmission order is even-odd-even etc. and as such the fields seems interleaved or interlaced. The North American and Japanese standard is called NTSC after the National Television System Committee and is likewise interlaced but with 480 apparent horizontal lines of video and 60 fields per second.[1] In their digital forms used on storage media like DVD Video and in digital broadcast, the resolutions are vertical by horizontal, PAL and SECAM: 576×720 pixels, and NTSC: 480×640.

[1]Nominally 60 Hz/100.1% $\approx$ 59.94 to avoid interference with the 60 Hz power source it was set to match originally. In the same way PAL and SECAM match 50 Hz AC power sources, but due to the different color systems used, only NTSC had serious interference problem. Still NTSC is popularly know to be short for Never The Same Color.

Most flat panel screens (Plasma, LCD plus some emerging new technologies), projectors (DLP, LCD etc.) and classic 'deep' PC monitors (cathode ray tube, CRT) are progressive, meaning they need deinterlacing – interlaced to progressive conversion – to be able to show interlaced PAL, SECAM and NTSC broadcasts or video. Furthermore most displays are of a higher spatial and/or temporal resolution and thus requiring video super resolution and/or temporal super resolution. The old standard definition (SD) formats are slowly being replaced by high definition (HD) formats, the dominating ones being 720p (720×1280 progressive, 24, 25, 30 or 50 fps) and 1080i (1080×1920 interlaced, 50 or 60 fps) or 1080p (1080×1920 progressive, 24, 25 or 30 fps). These formats are used in high definition television (HDTV) broadcasts and on the new digital video disc media HD DVD and Blu-ray disc.

PAL, SECAM and NTSC use the academy screen aspect ratio with a 4:3 = 1.33:1 frame width to height aspect ratio, whereas the HD formats are widescreen with a 16:9 aspect ratio. The academy format was the preferred film aspect ratio until the 1950s recommended originally by the American Academy of Motion Picture Art and Sciences, thus the name. As television took over the format, motion pictures gradually switched to widescreen formats, e.g. cinemascope. The HD pixels in widescreen are 1:1 in size, but with use of 1.422:1 pixels PAL also exists as an SD widescreen format, which is used for most cinematic moves on DVD.[2] The pixel squeeze used is similar to the format squeeze obtained when recording films with an anamorphic lens to get e.g. 2.35:1 cinemascope and thus denoted anamorphic widescreen.

Telecine is a process scanning recordings from film to video and even though they are displayed at 50 or 60 interlaced fields per second, they are recorded progressively at 24 frames per second (fps). Converting progressive source video back from interlaced is called a pulldown – 3:2 for NTSC and 2:2 for PAL and 30 fps film recordings (many US tv shows, e.g. sitcoms like *Friends*). Thus deinterlacing only denotes the process of actually creating extra lines in interlaced to progressive conversion.

All these facts on standards given above arise not only from technical and physical issues and limitations. The values and characteristics as for instance frame rates, are often fixed at their given values to accommodate properties of the human visual system (HVS), and interlacing was also employed to (ab)use properties of the HVS. To optimize the viewing experience, the THX norm (`www.thx.com`) was defined based on the minimum requirements needed to give a 'good' movie experience. Thus sound and video equipment and movie theaters upholding this norm gets THX certified as a sign of high technical quality.

2.2 Human Vision and the Displaying of Image Sequences

The reason why we are at all interested in doing upscaling is to enhance the viewing experience of human beings. There is no doubt that visual processing takes up a lot of the capacity of the human brain. Better visual inputs will give

[2]This pixel ratio is most often given as 1:1.422. There seems to be no general agreement on whether to use height:width or width:height when giving aspect, pixel, screen and frame ratios or sizes, but we will try to stick to using height:width in accordance with the rows, columns notation used for 2D matrices.

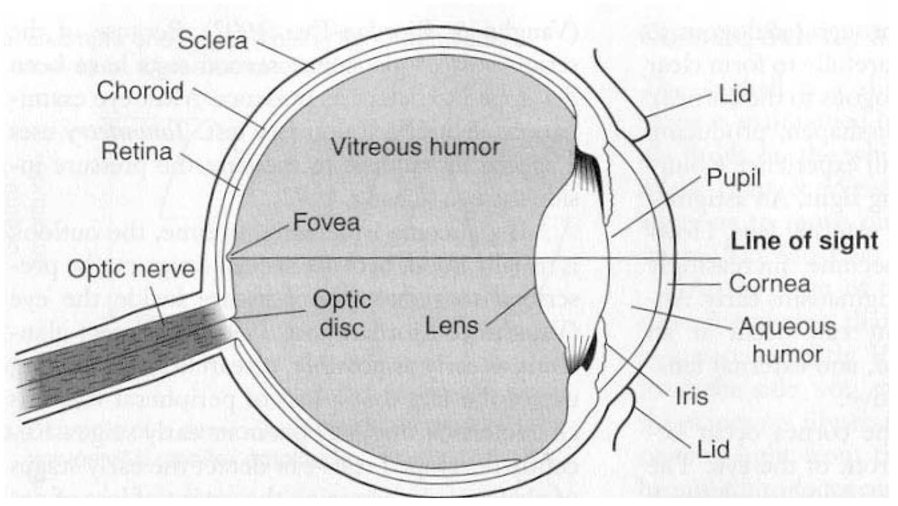


Figure 2.1: The human eye. Figure from [73].

two improvements. First, the better the input provided the less resources are spend processing it, which makes the viewing experience more relaxing. Second, the more natural the input appears, the less it will annoy the viewer. And that is what we are after.

Visually pleasing a human being can be done on several levels. On the highest level it is a question of psychology, does a person like certain colors more than other colors, does she prefer football over fashion shows, horror movies over comedies etc. All lower level vision serves to aid higher level vision. In providing quality inputs for the lower level vision, technology help film directors, tv producers etc. get their message through, e.g. by making sure the viewer sees colors the right way, that the image sequences displayed are sharp and detailed, that there is no annoying artifacts brought forth by bad recording, transmission, storage, coding etc. We do not want to change the artistic (or commercial) message presented by the sender, but we wants to make sure it reaches its human receiver as undistorted as possible, providing the lower level vision with optimal input. Unless references otherwise, the sources of any information given in the following subsections of this section (2.2) are the excellent book on human sensation and perception by Matlin and Foley [73], and to a lesser extent the book [43] by Gonzalez and Woods on digital image processing. Certain technical details mentioned are given in the references mention at the beginning of Section 2.1.

2.2.1 Basic Properties of the Human Visual System

The lower level part of the human visual system (HVS) consist of the eye, a visual pathway and (parts of) the visual cortex in the brain. A schematic drawing of the eye is shown in Figure 2.1. Light is projected through the lens and the cornea, ensuring (in the healthy eye) the information carried by the light reaches the retina covering the back of the inside of the eye. The retina is covered by photoreceptors called rods and cones. The rods gives us the ability to see when there is very little light, but only provides grey scale information. The widely used RGB color model is based over how the cones operate: There are three kinds of cones each covering parts of the visual light spectrum giving us color vision when there is an abundance of light available. The photoreceptors are connected vertically through bipolar cells to ganglion cells, each of the ganglions collecting the input from a number of photoreceptors forming a receptive field. It is the size of these receptive fields that crudely put decides the resolution of the eye and the HVS. But the density of receptive fields vary on the retina and already in the horizontal interconnection of photoreceptors and receptive fields through amacrine and horizontal cells complexity is added to the processing of the visual stimuli and the same goes for it passage through the visual pathway also doing some processing of the data. This illustrates how complex human vision is and explains why measuring and deciding on the properties of the HVS is not straightforward.

It is however a given fact that the stimuli of the receptive fields are treated in the visual cortex. On a higher level complex objects are recognized, but very interestingly the less complex processing in the visual cortex responds to edges, producing different signals according to the orientation and strength of the given edge. This makes edges a primary cue in vision. When there is no edges in the light signal projected onto the receptive fields in the retina, no signal is transmitted from the edge processing cells in the visual cortex connected to this receptive field. Thus we have to fill data from around edges into any smooth area where there is no edges present to give stimuli. Thus the variational image model total variation modelling images as smooth regions separated by edges mimics the lower level human vision rather well.

We have so far been concerned with the spatial processing of the HVS, but the time dimension also plays a role. A uniformly colored input (no edges seen in it) covering the full visual field will over time fade to a medium gray as there is no changes in the input to the eye over time and the receptive fields will just transmit a 'standby' or 'null' signal. It is changes in light the cells register and to keep the input from fading the eye makes small rapid movements all the time. When there are no edges no changes occur and thus the fading described above occurs, but only after a while (when both the memory stored and the filling-in effect described further above will fade out). A bit more important to our work is the fact that the eye to a certain extent is able to track and give the HVS a fixed and detailed view of moving scenes and objects. Another very interesting fact and the reason we have motion pictures is that the eye can be fooled to see motion where there is really none. Stroboscopic movement, e.g. created by flashing a light briefly at one location of the retina and shortly after flashing another light at another location will make the viewer perceive it as just one light moving in a straight line from one position to the other. The same sensation of movement is obtained with most rapidly changing patterns, e.g. still images only shown briefly to create motion pictures.

The eye is generally very sensitive to changes in stimuli (mostly induced by movement in the scene viewed) especially away from the center of focus on the retina (the fovea). The size of the area changing is also important to the response of the eye, the larger the area of change, the larger the sensitivity. The sensitivity to temporal changes in stimuli is called flicker sensitivity in the field of image sequence display.

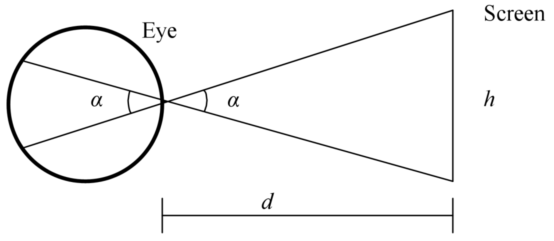


Figure 2.2: The projection of an on-screen image onto the retina. α is the vertical viewing angle dependant of the screen height h and the viewing distance d. The same relations are found for the horizontal (and diagonal) directions. The resolution of the screen ultimately set the limit of the maximal viewing angle possible.

2.2.2 Meeting the Requirements of the HVS in Technology

The many characteristics of the lower level HVS given above have been important in fixing the specifications of the video and broadcast standards described in section 2.1. Here we will discuss some the connections from HVS to technical specifications and requirements.

First, the spatial resolution of a discrete (digital) image stimuli has to meet the resolution of the HVS. The resolution of a screen is critical to how close viewers can be placed to the screen without spotting pixels instead of what appears to be a continuous image. The viewing distance and the size of the screen decides the viewing angle and the size of the projection of the image onto the retina. A sketch of this projection is given in Figure 2.2.

We know that a) pixels are square and receptive fields mostly circular, b) receptive fields are unevenly distributed on the retina and of different sizes, c) the visual processing and perception of details more complex than just defining the size of receptive fields, and d) the properties of the stimuli (brightness, color etc.) and viewing conditions affect the viewing experience. Thus the minimum required resolution of an image display at a given viewing angle has to fixed empirically. The THX norm requires horizontal viewing angles of at least 25° to probably ensure engulfment in the visual experience, keeping the viewer focused on the on-screen story and nothing else. Since degrees are not that easy to measure in the living room, the equivalent minimum viewing distance required is more widely used. On an standard PAL 576i tv-set the minimum viewing distance is six times the height of the screen. With the higher vertical resolution obtained by switching to progressive, a 576p screen should be viewed at distance of at least 4.3 times the height [108]. Thus deinterlacing increases the engulfment at a given screen size if it is done right (that is, deinterlacing without creation of visually annoying artifacts). Increasing the spatial resolution (VSR) will enable increased viewing angle and also lift the viewing experience if done right. The viewing distances given here and in [108] for other formats as well are not fixed as many factors decide the minimum and optimal viewing distances, but the viewing distance given serve as a frame of reference.

We have now established that increased spatial resolution is beneficial thus justifying the need for deinterlacing and video super resolution but also the temporal resolution is important.

The ability of the eye to induce apparent motion when exposed to rapidly changing patterns is used to make the *phi-effect* occur: Frame recordings (stills) of moving objects are shown in succession at a rate high enough to makes the motion depicted seem fluent. This is motion pictures. In early film recordings 16 fps was considered enough to create the phi-effect, but as technology evolved the film recording frame rate was since increased to 24 fps, which is the rate still used today. The rapid change of images needed to create motion pictures can (depending on the screen size and brightness etc.) be sensed by the eye as flickering, thus often requiring the frames to be updated faster than what is required to obtain the phi-effect. That is one of the major reasons why the PAL, SECAM and NTSC standards have field rates of 50 and 60 Hz.[3]

The area dependency of the flicker sensitivity in the eye described at the end of Section 2.2.1 is why interlacing exists. Each line can be updated at half the rate at which the full screen is updated without the eye sensing it – the area of a single line is too small to make it flicker in the eye in spite of it low refresh rate. But as screens grew larger and brighter (in living rooms not growing at the same speed) the flicker of single lines became apparent. With that a need of breaking the limits set by the SD television formats appeared. Before HDTV was introduced, PAL tv-sets with doubled field rates (100 Hz) where developed in a successful (but time limited) attempt to improve quality without changing the broadcasting standard.

The viewing angle dictated by the viewing distance (fairly static in most homes) and screen size is the major reason why higher refresh rates are needed, but also the screen type (CRT, LCD, plasma, film/DLP/LCD projector), screen brightness, image content and the general viewing conditions (lighting conditions in the room mainly) puts requirements on the refresh rate. Higher refresh rates can however be obtained by just redisplaying the same frame, which works fine on e.g. computer monitors as they mainly show stationary content.

In movie theaters with 24 fps recorded progressive on film (similarly on LCD screens) one can just let the current image be displayed constantly for 1/24 of a second as it will not flicker. The same goes in principle with e.g. LCD screens as they do not suffer from the same problem as CRT screens (and to some extent plasma screens): The image on screen will fade over (very short) time on a CRT and thus needs refreshing in order not to flicker. Projection of the same progressive frame or image for a longer time onto the retina – even just for 1/24 second in moderns cinemas – will make it appear to be a still image and thus give a sense an abrupt transition to the next frame and destroy the illusion of fluent motions otherwise obtained by the phi-effect. Therefore a shutter (a rotating disc which is semi hole, semi impenetrable to light) in the projector will give short blackouts once or twice during the display of a single frame, increasing the frame rate to 48 or 72 fps. The same need exist on LCD screen displaying video. Still, with only 24 fps recorded, high contrast edges in motion will sometimes appear as moving jerky thereby breaking the illusion of motion pictures. This failure to uphold the phi-effect also goes for video versions of film recordings and HD recordings at lower frame rate. Thus the need for increased temporal resolution is there.

[3]The values were beyond what was needed with early tv sets under normal viewing conditions, but the values were fixed at these specific values to correspond with the frequency of the electrical AC power supply to simplify the engineering of tv sets as mentioned earlier.

2.2.3 Good and Bad Human Vision

As mentioned above the retina photo receptor resolution is not decisive when it comes to the limit of detail perception in the HVS. Recent research by Rucci *et al.* [87] has confirmed what was long suspected: The rapid eye movements do not only serve the purpose of keeping visual stimuli from fading over time, it is also used to increase perceived resolution by integrating the low resolution input of the retina over time. This is human vision at its best.

The eye is able to track motion in an observed scene to a certain limit, and when this limit is crossed only motion blurred input is received on the retina. But in some cases the HVS fools its owner by deciding to classify some (motion blurred) input as sharp even though it is only registered in a blurred version on the retina, [15] and [74]. This could be seen as a counterpoint to the case of good vision described just above. In the space between the two cases we need to provide sharp an detailed video input at a reasonable computational cost. The challenge will be to produce as sharp and detailed video as possible and keep it free of any artifacts. We must try to remove artifacts present in the input and not introduce new ones in the upscaling process. In the next section we will discuss how to measure the quality of our upscaling results.

2.3 Quality Assessment: Objective vs. Subjective Measures

”Do you like fermented beans?” Ask a person this and many will not know what fermented beans are and among those who know, mostly Japanese people will answer yes as it is a traditional Japanese dish. Taste, emotions, feelings, inheritance and environment; humans are formed and driven by many factors hard to measure as objectively as a number, we are individual beings with subjective opinions and tastes. When it comes to vision there might be similar measurable neural responses in lower level vision to a given visual input, but we can unfortunately not use these lower level patterns to predict how each individual judge or psychologically interpret the visual stimuli leading to this specific neural pattern. The human psyche and the brain as such are not (yet) mapped to a degree enabling quantification of it, and we are anyway highly subjective in our judging of visual – and other – inputs.

Using statistics on a larger number of subjective evaluations done by human observers can of course give some average answers on e.g. the quality of a given video enhancement method. But statistically significant answers require a large test panel and a standardized test setup like the ITU[4] recommendation for subjective quality assessment of video in [50]. The Danish producer of audiovisual consumer electronics Bang & Olufsen has its own setup using both layman and expert panels [100] and the same goes for Philips in the same line of business as described in [4]. These large scale subjective evaluation setups are so time and resource consuming that they are often not used when presenting new video processing methods. The alternatives are either smaller subjective test panels consisting of the authors and maybe some colleagues, or (simple) objective measures – or often both. Typically results are compared to results

[4]ITU is the International Telecommunication Union, a United Nations agency.

obtained by existing methods, simple or advanced, to show the qualities of the new method presented – in the good cases the visual differences are very clear and thus the advantages of the new methods clearly shown.

Objective measures give clear cut differences between results obtained with different methods and leaves no room for interpretation, but are also often very far from giving a true picture of how the results would be evaluated by a large group of subjective viewers. Objective measures require a ground truth for comparison and thus we artificially need to create the degraded low resolution video by removing (scan) lines, downsampling frames, or removing frames. This will of course only serve as a simulation of real 'degraded' data (e.g. video recorded with an interlaced camera) and thus devaluate the use of objective measures. Exceptions from the requirements of a ground truth sequence are the MTI and MSE_i measures for deinterlacing evaluation given by Bellers and de Haan in [4], but the measures then depend on the optical flow computed on the sequence. The two most widely used objective measures are: Mean square error (MSE) and peak signal to noise ratio (PSNR). We have used objective evaluation with varying success. For assessment of deinterlacing (Chapter 3) we show how the MSE does not correspond with the subjective evaluation, whereas it corresponds perfectly in video super resolution evaluation (Chapter 4) and temporal super resolution evaluation (Chapter 4). For the case of deinterlacing a study of quality assessment carried out by Zhao and de Haan in [109] compares PSNR with an 18 person subjective evaluation finding statistically significant correlation between the two. This study can along with ours be considered limited cases, as the data sets are small (in [109] only five test sequences are used and the test data is also subject to some postprocessing before the subjective evaluation is carried out).

In our opinion objective evaluation gives a hint of what to expect from a subjective evaluation, but should never be used on its own, this goes also for more advanced objective measures that claim to model the HVS closely. 36 objective measures are put to the test by Kanters in [51] to evaluate image reconstruction, and it is *"...concluded that it is almost impossible to find an objective error measure that correctly resembles the human observer results."* ([51], Chapter 4, p. 76.) We agree on this and are of the opinion that in limited cases (small data set or certain tasks, e.g. video super resolution) some objective measures might actually agree quite well with the subjective evaluation, an opinion that it is also uttered by Kanters in [51] and Nadenau *et al.* in [77]. In the latter Nadenau *et al.* summarizes the results of a larger test of subjective (following [50]) and objective video quality evaluation by the Video Quality Experts Group (VQEG) [106]. The correlation in quality ratings between subjective evaluations conducted at different labs is 0.9 - 0.95 while the best objective measures only have correlations of 0.8 - 0.85. In [84] Puttenstein *et al.* have tested 20 objective quality measures in evaluating noise reduction in video, and found none of them to correlate very well with the subjective evaluations conducted as well.

In the conclusion of [84] Puttenstein *et al.* writes: *"Even a combination of objective measures only approximates the subjective assessment of the quality of noise reduction algorithms with a descriptive power as low as 59%."* In the conclusion of the report by the Video Quality Experts Group [106] it is stated that the VQEG does not recommend the addition of any objective measure to the ITU Recommendations on video quality assessment. Furthermore it is stated that no objective measure is able to replace subjective evaluation, a conclusion we had also drawn before knowing of this report.

The objective measure used most widely for assessment of accuracy in optical flow calculations, the average angular error (AAE) suffers from the same problems as the objective image quality assessment measures and can only be used on artificial sequences due to the lack of a ground truth optical flows on real world sequences.

2.4 Color Video Processing

All inputs and results given in this thesis are 8-bit luminance channel video. There are several reasons why we have not produced color video results. First, and least important is that in many publications color printing is not an option or rather expensive and thus it is customary to present gray scale results only. Secondly, extension to full RGB or YC_rC_b is straightforward as all channels can be processed independently using the same code as applied to just the luminance channel. More advanced vectorized version of our algorithm could be devised (e.g. following the directions given in[103]) which should give better coherence in geometrical structures between the three channels and thus better results. This leads us to the third reason why we do luminance processing only: Advanced vectorized color processing will most likely not be beneficial on video in general. Analog video is broadcasted and stored using half of the total bandwidth available for the luminance channel, sharing the other half between the two color channels [70]. This has two reasons, first, using YC_rC_b and not RGB made it easy to add color information to the black and white broadcasting signals so that people were not forced to buy new color tv sets and the broadcasters not forced to broadcast both in color and black and white. At the same time the subsampling of the color channels made room for more broadcast channels. Secondly, the HVS is less sensitive to information (on edges) given in the color channels than information given in the luminance channel. The color compression by subsampling w.r.t. to the full frame resolution has also found its way into digital video and broadcast where MPEG-2 used on DVDs and its HD successors (MPEG-4/H.264) all subsample the color channels when applied in practice on video. The term 4:2:2 describes the sampling ratios of the YC_rC_b signal and along with 4:1:1 (or 4:2:0, see [107] for details) this factor two subsampling in one or two dimensions are the commonly used color formats.

Therefore it is obvious to just apply simpler methods for the color channels in deinterlacing. As described in Chapter 3, throwing the full variational algorithm at the color channels is a complete waste of resources. For temporal super resolution where fully new frames are created it will of course be necessary to fully process the color channels, but since the resolution of each color channel is 50 or 25% of that of the luminance channel and the flow will be almost directly reusable as the channels are highly correlated, the computational overhead will be small. For video super resolution simple interpolation could be used on the color channels as long as the magnification factors are not to large, but when the full algorithm needs to be applied, the flow can again be reused.

2.5 Optical Flow

In this section we will give a short introduction to the importance of knowing how an image sequence changes from one frame to the next, how we find the optical flow of the image sequence. Focus will be on how we can use optical flow computations to improve upscaling by doing motion compensation (MC).

2.5.1 Temporal Information and Motion Compensation

It should be clear by now, that temporal information is of high value in video processing. In the 2D spatial plane of an image or video frame, we have in each point access to neighboring information, helping us to improve processing. The information could be knowledge of edges, how strong are they and what are their orientations, helping us to preserve and maybe even enhance the edges when processing the frame. There is an abundance of information available in image sequences due to the (relatively) dense temporal information sampling. If there is no or only very small motion (less than one pixel in size in the digital/discrete image sequence grid) the relevant temporal information correlated to the pixel currently being processed is easily found. In case of larger motion we need to follow the trajectory of the motion to find the relevant temporal information: We do motion compensation, which requires an estimation of the motion first.

Motion compensation is a cornerstone of all our upscalings and in any image sequence with motion it is crucial to now the flow if one wants to optimize the output quality. But there is a price to pay for the gain in quality: Estimating reliable and precise motion is a complex task.

2.5.2 Motion and Optical Flow

The physical world is 3+1D (3D space and 1D time). In a projection of the world into 2+1D as done when recording image sequences (2D spatial image frames and time) information will be lost, and thus we can only do an estimation of the real, physical 3D motion. The intensities (or in many cases, color) and positions projected onto the 2D frame plane is all the information we have available when estimating the motion in image sequence analysis and processing – but also as human viewers of film, video and television. The apparent motion perceived when viewing or analyzing 2+1D image sequences is in the computer vision literature called *optical flow*.[5] Optical flow does in some cases differ from the true projected 2D motion, the textbook example being the zero optical flow in a projection of a rotating, uniformly colored sphere, the true projected 2D motion being the rotation. An illustration of this example is given in Figure 2.3. The optical flow maps the changes from one frame into the next, which is what we would like to compute when doing motion compensated video upscaling. Any knowledge of the geometrically true 2D motion or for that matter the physically true 3D motion, will not improve the upscaling and thus the data we transmit to the eye via the display as it is still just an optical 2D projection. In other fields of computer vision, e.g. robot navigation, true 3D motion reconstruction is of great importance.

[5]We will use the terms optical flow computation/calculation/estimation and motion estimation interchangeably from here on, as we do estimate the motion when computing the optical flow.

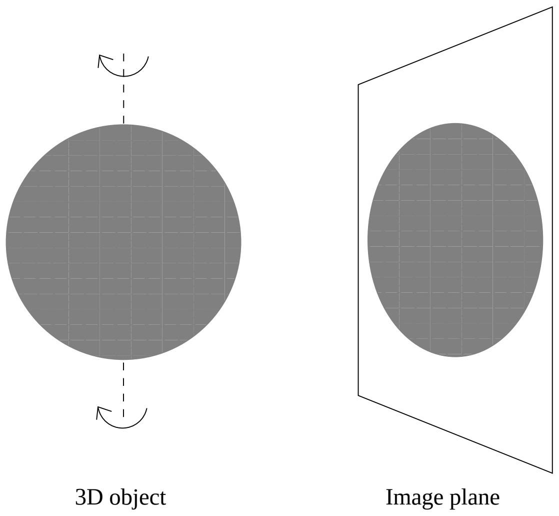


Figure 2.3: A uniformly colored sphere rotating around any of its own axes on the left and its projection onto the image plane on the right. Any 2D rotation (speed and axis) can be projected but the optical flow either computed or perceived will be zero unless additional information is provided. Figure from [64].

The foundation of optical flow calculation is the assumption that one can find information recorded in one frame in the next frame as well even if objects in the recorded scene moves around. We are looking for a displaced frame difference (DFD). It is typically assumed that the information we are looking for is (pattern of) intensities that do not change from frame to frame. Thus the DFD is also known as the brightness (or grey level) constancy assumption (BCA) and is

$$u(\mathbf{x} + \vec{v}, t + 1) - u(\mathbf{x}, t) = 0 \tag{2.1}$$

where $u = u(\mathbf{x}, t)$ is the 2+1D image sequence with $\mathbf{x} = (x, y)$ being the spatial coordinates and t the time dimension. The optical flow we are looking for is $\vec{v} = (v_1, v_2)$ where (v_1, v_2) are the (x, y)-coordinates of the flow from frame t to frame $t + 1$ assuming a distance of 1 between neighboring frames.

The intensities (or colors) cannot be expected to stay the same from frame to frame: A red ball is also red in the next frame no matter if it moves or not, but not if the lighting of the scene changes. The gradient constancy assumption (GCA)

$$\nabla u(\mathbf{x} + \vec{v}, t + 1) - \nabla u(\mathbf{x}, t) = 0 \tag{2.2}$$

which assumes the image gradient ∇u stays constant, helps to cope with the instability of the BCA under changing lighting conditions. The gradient of the edge between white an black patches of a football stays (close to) constant even if the light on the ball changes.[6]

[6]Gradients are the almost salient measure of edges in images and as we know from Section 2.2, edges is a key cue to the lower level vision. Thus computing optical flows with good gradient/edge mapping will be a help in doing good motion compensation.

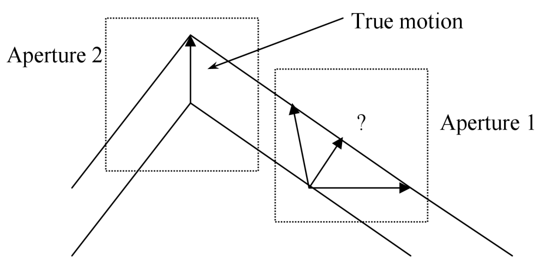


Figure 2.4: The aperture problem in optical flow computation. Two edges and corner moves from frame t to frame $t+1$. In Aperture 1 we cannot uniquely identify the correct flow, but in Aperture 2 we have enough (local) information to uniquely determine the correct flow.

When computing optical flows one would like to get it very detailed and thus use a small aperture in which we define the flow for each part of a frame. The aperture is often a small block (e.g. 8x8 pixels) or single pixels in discrete settings. But when we use a small aperture, we do not always get all the information we need to calculate the flow correctly. The general problem of only having very local knowledge of what is going on is called *the aperture problem*. Say we have a line moving, then any point on the line in frame t could go to any point on the line in the next frame $t+1$ as illustrated with 'Aperture 1' in Figure 2.4. Say we extend our aperture to include a corner at the end of the line, then in the corner point we have spatial gradient in two direction and can determine the exact flow for the whole line as with 'Aperture 2' in Figure 2.4. The second fundamental problem in computing optical flows is the *occlusion problem*. When an object moves it covers (occludes) and uncovers (disoccludes) background or other objects behind it in the scene. Since we are looking for similarities between frame we have a problem when information is only present in one of the frames. A solution to the problem when one needs the flow to do motion compensated video processing, is to look for information along the flow both backwards and forwards in time.

In the next section we will look at how we solve the aperture problem and handle the occlusion when using variational methods to compute optical flows.

2.5.3 Variational Optical Flow

Variational methods for optical flow calculations are among the most accurate in existence as found in the major survey by Bruhn *et al.* in [12] and they fit perfectly into our variational framework presented later in this chapter. A very detailed work on variational optical flow is the PhD thesis by Bruhn [13].

All three of our upscalings (DI, VSR and TSR) are motion compensated thus they need an optical flow field to be operational.[7] This complicates matter as typical implementations of variational optical flow methods can be computationally heavy. However, being able to draw on temporal information from neighboring frames is a major gain and will provide improved quality (e.g. shown for the case of inpainting in [64] by Lauze). Returning to the computational

[7]We also present a motion adaptive deinterlacing algorithm but our focus is on the motion compensated deinterlacer.

complexity of variational optical flow methods, it is possible to run them in realtime. In [11] Bruhn *et al.* presents an implementation of a linear, variational flow algorithm that produces dense web-cam resolution (18 frames per second (fps), 252×316 spatial resolution) flows in realtime on a standard PC. The nonlinear methods we used are (a lot) more complex and even SD video (PAL, 25 fps, 576×720) requires more resources, but with dedicated hardware and the right amount of parallel processors it should not be impossible as we will discuss in Chapter 7 of this thesis.

Fitting any variational optical flow methods into our framework given later in Section 2.6, is straightforward. Practically all variational flow methods from the early method of Horn and Schunck in [48] to the so far most accurate method by Brox *et al.* presented in [9] consists of two terms. The first is a data fidelity term defining the temporal correspondence from frame to frame and is derived from the brightness and gradient constancy assumptions in (2.1) and (2.2). The second term is a prior on the flow telling us about the expected local correlation between neighboring flow values. In short the prior models the fact that local clusters of pixels will have the same or very similar motion as they will represent just one object with (relatively) uniform motion due to the physical rigidity of the world. It is more probable that neighboring pixels will have the same flow than have very different flow. The prior is also the key to solving the aperture problem, although we really do not care where on the line in Aperture 1 in Figure 2.4 we get our information from when using the flow for motion compensation. Still, the prior term imposes a necessary regularization on the optical flow computation not provided by the data term.

In the pioneer work by Horn and Schunck [48] the data term is the classical optical flow constraint (OFC),

$$\vec{v} \cdot \nabla u + u_t = 0 \tag{2.3}$$

where $\nabla u = (\partial u/\partial x, \partial u \partial y)^T$ and u_t is the derivative of u w.r.t. time. Equation (2.3) is the differential and linearized version of the brightness constancy assumption in (2.1). The OFC can also be written

$$\vec{v} \cdot \nabla u + u_t = \vec{V}^T \nabla_3 u = \pounds_{\vec{V}} u = 0$$

where ∇_3 is the spatiotemporal gradient of u, $\vec{V} = (\vec{v}^T, 1)^T$ and $\pounds_{\vec{V}} u$ denotes the OFC as the Lie-derivative of u along the flow $\vec{V}$ (see e.g. [42]).

In [9] Brox *et al.* combines the BCA in the data term with the gradient constancy assumption, which in its linearized form is

$$\pounds_{\vec{V}} \nabla u = 0 \tag{2.4}$$

the $\vec{V}$-directional derivative of the spatial image (frame) gradient.

Taking the classical approach to variational optical flow computations, the prior term is added to the BCA data term as the aperture problem is an analytical derived consequence of having one equation (the BCA data term) with two unknowns, v_1 and v_2 of the flow vector $\vec{v} = (v_1, v_2)$. But still with the GCA added to give us the extra equation, the regularization imposed by the prior (smoothness term) is important to get a dense flow field by pushing flow values into areas where salient image content is missing (e.g. along the line in

Aperture 1 of Figure 2.4. If the line from Aperture 1 to the corner in Aperture 2 is long, the aperture will be too small even with the prior added. The well-established solution to this problem is to first scale down the sequence and compute an initial guess of the flow at a coarser version of the image sequence. This is known as the multiresolution approach when it is done in several steps forming a pyramid of scales. Since the optical flow constraint and the linearized GCA are implemented as filter with only very local support, it will take many iterations to compute flows of high magnitude. Using multiresolution will also scale down the magnitude of the flow and thus also aid computation of high magnitude flows.

Between the landmark works by Horn and Schunck [48] and Brox *et al.* [9] many variations of variational optical flow have been suggested. For in-depth reading on variational optical flow we refer to the PhD theses by Bruhn [13] and Lauze [64]. We ended up mainly using the method of Brox *et al.* [9] as it is the most accurate method and had also been used successfully for motion compensated inpainting by Lauze and Nielsen [65].

The use of this method (and other variational approaches) on real world video has been relatively limited. When evaluating the quality of a method, the goal is to minimize the angular error to the ground truth flow, which only exist if the test sequence has been artificially generated like the `Yosemite` sequence mostly used in optical flow benchmarking. Thus we conducted a tedious but also thorough test to optimize the settings in our implementation of the method from [9] (on both interlaced and progressive video) by visually judging the quality of the flow fields obtained using different parameter settings on real world data. The optimal parameter settings are the ones presented in each of the Chapters 3, 4 and 5 of this thesis.

2.6 Variational Upscaling Framework

Our upscaling problems are sampling problems. Deinterlacing and video super resolution are subsampling problems as the number of recorded spatial samples is too low compared to what we need. Temporal super resolution is in most cases also a subsampling problem, but can also be a case of upsampling. As we will make clear in Chapter 5 on temporal super resolution, creating fully new frames is a hard problem no matter whether you increase or decrease the frame rate of a sequence.

We will start our problem analysis using the sampling theory on the subsampling problem. In the recording process the 2+1D data projected from the real world has been sampled at frequencies lower than what we would like to display it at. The Nyquist-Shannon (sampling) theorem

$$F_{max} \leq \frac{F_s}{2}$$

tells us to use a sampling frequency F_s at least twice as high as the highest frequency F_{max} we want to be able to sample from our signal. In video this decides how fine details of the scene depicted we will be able to include. Frequencies higher than $F_s/2$ will be aliased if present in the signal and no (analog) low pass filtering is applied prior to the sampling (for details refer to [83] and [43]).

The essential message we get from this analysis is that our upscaling problems as they are subsampling problems by birth, are ill-posed. We will never be able to create what was never sampled, the sub-sampling process is an irreversible problem. Due to the presence of motion temporal super resolution is also an ill-posed problem when decreasing the frame rate. Spatiotemporal frequency analysis of the deinterlacing problem in the presence of motion has conducted by Bellers and de Haan in [4].

2.6.1 Bayesian Inference

To model image sequences, their optical flow and content and to impose some regularity to the ill-posedness we need an inference machine. The human viewer might be satisfied with many different solutions to our problems, but even if the human observer cannot see when or if the sampling process has been reversed, trying to go back as far as we can towards the analytically true solution is a good idea. We know it will be satisfactory for sure and where we cannot get close, the regularity will help us not stray to far in a wrong direction. Especially if the model applied for regularization lies close to how the human visual system itself imposes regularity and generally processes visual input. Exemplified the above problem view means that going towards how an standard definition signal would have looked recorded in high definition (HD) is a good idea when doing upscaling. Doing so we should impose regularity in our model to avoid creating new artifacts (e.g. the ringing patterns often created by super resolution algorithms) that would annoy the viewer.

The inference machine we will use to define a framework for our upscaling problem of creating new data in an image sequence and impose some regularization on the ill-posedness of the problem is the Bayesian inference

$$p(a|b) = \frac{p(b|a)p(a)}{p(b)} \tag{2.5}$$

where $p(a|b)$ is the a posteriori, $p(b|a)$ is the likelihood, $p(a)$ the prior on (or simply the probability of) a and $p(b)$ is the evidence of b.

When it comes to the need of (re)creating information never sampled or missing due to a degradation of the image sequence data, de-noising, inpainting and upscaling are very similar problems. The similarity becomes even clearer when defining a joint framework of the problems using Bayes inference. Substituting a with u a desired output image (de-noised, upscaled or inpainted) and b with u_0 the input image (e.g. noisy, of low resolution or scratched), we get

$$p(u|u_0) = \frac{p(u_0|u)p(u)}{p(u_0)}. \tag{2.6}$$

To keep this example clear, we have started looking just at single images. The likelihood $p(u_0|u)$ is also known as the data (fidelity) term in variational formulations as it tells us how the input relates to the output, e.g. how to place the existing lines of an interlaced signal in the deinterlaced, progressive output. $p(u)$ is the spatial prior on u, which decides what image content we allow in our images, how we model images. Maximizing the a posteriori, equal to maximizing the right hand side in (2.6), optimizes the probability of u knowing u_0, which is just what we wish to do. The process is called maximum a posteriori

(MAP) and to do MAP we do not need the evidence $p(u_0)$, which is just a normalization factor (the probability of the input is fixed since it is known), thus we have

$$p(u|u_0) \propto p(u_0|u)p(u). \tag{2.7}$$

The locus of missing data D which is always known in upscaling, can be harder to find when doing inpainting. We wish to include it as known in our model, that is we get a new a posteriori, $p(u|u_0, D)$ and from that a new likelihood term $p(u_0|u, D)$ and a new prior $p(u|D)$. Our choice of prior on u is independent of the locus of missing data, it models what (ideal) image sequences are, what we strive for. Thus $p(u|D) = p(u)$ and we have

$$p(u|u_0, D) \propto p(u_0|u, D)p(u).$$

In (pure) de-nosing D will just cover the whole image and thus can be excluded. The same goes for video super resolution, but in temporal super resolution and deinterlacing D tells us where data is missing. The likelihood term is what has to be different between different uses of our framework, thus it will be thoroughly discussed in each of the Chapters 3, 4 and 5.

Extending our formulation to be on image sequences instead of images, u_0 denotes an input image sequence and u a desired output image sequence. We now also wish to compute the flow, $\vec{v}$ of u, which gives us a new a posteriori, $p(u, \vec{v}|u_0, D)$, and thus get a new likelihood term, $p(u_0|u, \vec{v}, D)$ and a new prior, $p(u, \vec{v}) = p(u|\vec{v})p(\vec{v})$. Since the degradation or subsampling of u_0 is independent of the optical flow $\vec{v}$, the likelihood is $p(u_0|u, \vec{v}, D) = p(u_0|u, D)$. We have already discussed in Section 2.5 how spatiotemporal image sequences are not a 3D but a 2+1D volume with space and time being separate dimension, thus we assume that the $p(u|\vec{v})$ part of the prior is $p(u_s, u_t|\vec{v})$ and factor it as

$$p(u_s, u_t|\vec{v}) = p(u_t|u_s, \vec{v})p(u_s|\vec{v})$$

where the first term models the temporal coherence of an image sequence. The latter term on the right hand side gives a change to model motion blur, but as we consider motion blur a desired artistic expression and for the sake of simplicity we assume independence of u_s and $\vec{v}$, thus $p(u_s|\vec{v}) = p(u_s)$.

Gathering the new terms for image sequences, we end up at the Bayesian framework for joint image sequence inpainting and motion recovery given by Lauze in [64] (and Lauze/Nielsen in [65])

$$p(u, \vec{v}|u_0, D) \propto \underbrace{p(u_0|u, D)}_{P_0}\,\underbrace{p(u_s)}_{P_1}\,\underbrace{p(u_t|u_s, \vec{v})}_{P_2}\,\underbrace{p(\vec{v})}_{P_3} \tag{2.8}$$

which we will also use for deinterlacing and temporal super resolution. Leaving D out we have the form of the framework we will use for video super resolution

$$p(u, \vec{v}|u_0) \propto \underbrace{p(u_0|u)}_{P_0}\,\underbrace{p(u_s)}_{P_1}\,\underbrace{p(u_t|u_s, \vec{v})}_{P_2}\,\underbrace{p(\vec{v})}_{P_3}. \tag{2.9}$$

For both forms, (2.8) and (2.9), the right hand side terms are: P_0, the image sequence likelihood, P_1 the spatial prior on image sequences, P_3 the prior on motion fields as discussed in Section 2.5, and P_2 a term that acts both as spatiotemporal prior on the image sequence and as likelihood term for the motion field, which was also discussed in Section 2.5.

2.6.2 Variational Formulation

The Bayesian modelling framework could be used directly to maximize the a posteriori (MAP) but we rewrite (2.9) into an energy minimization problem. From [75] by Mumford we know that $E(x) = -\log p(x)$ (given that the probability functions are Gibbsian) and get

$$E(u, \vec{v}) = \underbrace{E(u_0, u)}_{E_0} + \underbrace{E(u_s)}_{E_1} + \underbrace{E(u_t, u_s, \vec{v})}_{E_2} + \underbrace{E(\vec{v})}_{E_3}. \tag{2.10}$$

Using calculus of variation we find and solve corresponding Euler-Lagrange equations as will be described in Chapters 3–5.

In deinterlacing the data term E_0 tells us where the missing lines are located in the spatiotemporal data volume that constitutes u, and it controls whether or not to do any processing of the known lines in the output. In video super resolution E_0 represents the super resolution constraint telling us how to map image data between high and low resolution frames. In temporal super resolution it ensures that we keep input frames untouched, also when they are located in the same temporal position as an output frame and thus included in the output (e.g. every other frame in the output of a frame doubling).[8]

The data term is the only term that needs to be different in different applications of our framework, the remaining terms could each be exactly the same no matter what kind of upscaling (or other video processing) we were doing. They define what image sequences and their flows as the should be according to the model chosen. We need to chose these terms to give the best possible modelling of real image sequences, but still keep the implementations of the framework mathematically tractable and the computational complexity down to a reasonable level. Total variation is mostly a good compromise between optimal output quality and complexity.

The last term, E_3 is the regularization on the flow (equivalent to the prior P_2). We have discussed in Section 2.5 how total variation will give better flow segmentation as it preserves edges. For the spatial regularization of the intensities E_1, it is well-known from image de-nosing and diffusion that total variation will help preserve edges. For the E_2-term total variation will help handle occlusions in the flow calculations as it was also mentioned in Section 2.5. For the intensities the beneficial effect is even higher as total variation in E_2 will stop diffusion across temporal edges at occlusions.

Actually minimizing (2.10) is done by minimizing two separate, yet coupled equations, $E(u)$ and $E(\vec{v})$. As they are functionals of only u or $\vec{v}$, the terms E_0 and E_1 only appears in $E(u)$ and E_3 only in $E(\vec{v})$. But E_2 is a mixed term, but we can still formulate it differently in $E(u)$ and $E(\vec{v})$. This will to some degree ruin the beautiful theoretical consistency of our framework, but will in practice be a tool to optimize results and control complexity. In general we use both the brightness constancy assumption and the gradient constancy assumption when minimizing the energy of the flow $E(\vec{v})$ as it is done by Brox *et al.* in [9]. For the intensity energy $E(u)$ the GCA will lead to a fourth order term

[8] In temporal super resolution we consider every frame a discrete time sample with near zero temporal aperture, but in case one wishes to model the temporal aperture of frames, then there could be overlaps between output and input temporal point spread functions. In that case one needs a more complex data term close to the one used for video super resolution, but with added modelling for temporal dynamic events (motion).

in the Euler-Lagrange equation derived from it (see [64]). Since the intensity Euler-Lagrange equation otherwise only contains first and second order terms, adding the GCA will make the minimization of $E(u)$ a lot heavier. Since we use Taylor approximations to get numerical implementations of the Euler-Lagrange equations and we solve them by taking small, iterative steps, this fourth order term will (most likely) have very little effect on the overall result, and thus we leave it out of $E(u)$. Further discussion on the (possible) use of the GCA in $E(u)$ can be found in Chapters 3– 5.

A very important feature of our framework is that it suggests simultaneous optimization of the flow and intensities. The idea is that the solution to u will be optimal if we know the optimal flow $\vec{v}$ and vice versa. True simultaneousness is a theoretic concept, but we get as close as possible in practice by solving this chicken-egg problem incrementally, switching between improving u and $\vec{v}$ as we iterate towards a solution. Thus we stepwise improve both flow and intensities when minimizing $E(u)$ and $E(\vec{v})$ in parallel.

Chapter 3

Deinterlacing

We present a variational framework for deinterlacing that was born for inpainting and since redeveloped for deinterlacing. From the framework we derive a motion adaptive (MA) deinterlacer and a motion compensated (MC) deinterlacer and test them together with a selection of known deinterlacers. To illustrate the need for MC deinterlacing the interlacing problem is introduced. It cannot be solved by MA deinterlacers or any simpler deinterlacers but only by MC deinterlacers. The major hurdle in doing MC deinterlacing is doing reliable optical flow computations (a.k.a. motion estimation, ME) on interlaced video.[1] We discuss a number of strategies for computing optical flows on interlaced video hoping to shed some light on this problem. We produce results on real world video data with our variational MC deinterlacer that even in many difficult cases are indistinguishable from the ground truth and in general the best deinterlacing results we have seen.

3.1 Introduction

Throughout the history of television interlaced scan has been the dominant scan format for recording, broadcasting, storing and displaying of video and television. Interlacing is cost efficient and has until recently been sufficient to ensure the best possible viewing experience to the human visual system (HVS). This is no longer the case for several reasons. First, interlacing artifacts that were once not a problem, has started to become visible as screens has grown larger and brighter with higher contrasts. Secondly, progressive scan screens (and tv/video cameras) has become available, offering significantly better viewing quality.

Progressive scan is when all horizontal lines of a 2D spatial video frame is scanned, whereas interlaced scan is alternating between recording only the even or the odd horizontal lines of the frames, denoting these half frames even and odd *fields*. Interlacing scan is illustrated in Figure 3.1(a). When the interlaced fields are recorded separated in time, then two neighboring fields (even, odd pair) cannot be merged to one full frame to be displayed on progressive screens without problems, and neither can the illusion of full frames being shown on

[1]In this chapter we will use the terms optical flow calculations and motion estimation (ME) interchangeably referring to the determining optical flow changing on video frame/field into the next frame/field as discussed in Section 2.5 of this thesis.

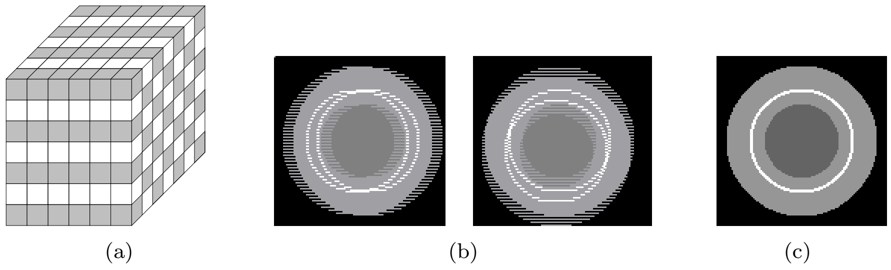


Figure 3.1: (a) The interlacing (width, height, time)-volume. (b) Interlacing artifacts shown by a merge of a time t field with its neighboring time $t + 1$ field, first serration due to horizontal motion in the scene is depicted, second line crawl due to vertical motion. (c) shows how the simple scene in (b) should look perfectly deinterlaced.

interlaced screens be kept when these interlaced screens are big and bright.[2] It is a well-known fact that the HVS is more sensitive to flicker in large areas (see Section 2.2.1), which is why interlaced scan was thought of in the first place: On smaller screens no one will spot single lines. It was as screens grew in size, brightness and contrast that single image lines grew big enough to be spotted individually by the HVS and interlacing artifacts became visible.

The interlacing artifacts serration and line crawl are shown in Figure 3.1(b), whereas the third kind, line flicker cannot be shown in print. It appears when a detail appears in just one horizontal line (it has the highest possible sampling frequency according to the Nyquist-Shannon sampling theorem) and is then only displayed at half the intended screen refresh rate and thus appears to flicker. Details on interlacing artifacts can be found in [4], [55] and [81].

The broadcasting and distribution of television and video media is still dominated by interlaced scan. Even with the rise of high-definition television (HDTV) and the known superiority in viewing quality of progressive video over interlaced video, interlacing persists, e.g. as one of the formats used for HDTV in the US and Japan (1080i, 1080×1920 resolution with only 540 lines scanned in each frame). As most plasma and LCD screens are progressive displays and they are dominating the world market of both tv sets and computer monitors today, and as devices used for tv and video (and music) are integrating with computers into media centers, there is – and will be in the future – a need for conversion between interlaced and progressive formats.[3] going from progressive to interlaced scan is a simple downsampling whereas interlaced to progressive conversion, more commonly known as *deinterlacing*, is an ill-posed upscaling problem.

The goal of deinterlacing is to create high quality progressive video from interlaced video, thus removing all interlacing artifacts, while doubling the amount of information as each field is converted to a frame.

In this chapter we will present a variational framework for deinterlacing and two deinterlacing algorithms developed from that, both of which do high qual-

[2]The size of the screen is of course only a problem if the viewing distance is short and thus the viewing angle big. Details on this can be found in section 2.2 of this thesis.

[3]The ALiS panel technology by Hitachi and Fujitsu is a known example of interlaced scan applied to flat panel displays.

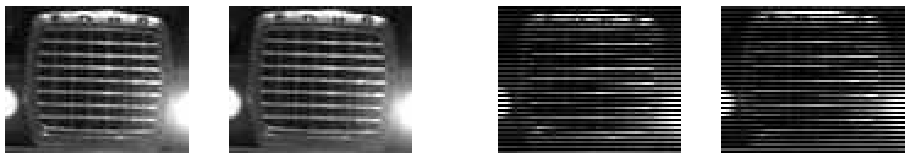


Figure 3.2: The Interlacing Problem. (a) Cutout of two consecutive progressive frames of the sequence `Truck` of a truck driving towards the camera. (b) the same two cutouts viewed as interlaced fields (by setting every other line to zero/black) where almost all information on the correct spatial structure is gone, thus making deinterlacing an extremely difficult task.

ity deinterlacing. Let us say we have a representative selection of interlaced video w.r.t. motion and details depicted (high spatial frequency content). Using simple spatial or temporal deinterlacing techniques on this video will yield acceptable (spatial interpolation) to poor (temporal interpolation) results and have plenty of interlacing artifacts left in the output. The poor quality comes from not adapting to or compensating for the presence of motion.

Using motion adaptive deinterlacers like one of the two we will present in this paper along with the ones given in [94] and in state of the art chips sets like the DCDi® by Faroudja (and Genesis Microchips) will often get rid of most of the interlacing artifacts.[4] Motion adaptive deinterlacers use motion detection to determine whether to use local temporal interpolation – in case of no motion and thus high local temporal coherence – and when to interpolate spatially only – in case of motion. Motion adaptive deinterlacers are, however, unable to solve what we like to call *The Interlacing Problem.* It is basically the problem of correctly deinterlacing sequences containing details (high spatial frequencies) in motion as in the example given in Figure 3.2. 'Correctly deinterlacing' meaning that the output appears detailed to the human visual system, and does not contain any visible artifacts to annoy it.

The Interlacing Problem can be handled by smoothing the region with details in motion to remove any interlacing artifacts but the resulting blur is also an artifact. A better choice is to propagate information along the motion trajectory (optical flow) doing motion compensated (MC) deinterlacing, which done right will transport detailed video content recorded in neighboring fields into the lines in the present field where they are missing. From a mere visual inspection of the two frames/fields of the sequence `Truck` given in Figure 3.2 it is obvious how difficult it would be to recreate the correct structure using only local temporal and spatial information as all non-MC deinterlacers do. Figure 3.3 shows another example of The Interlacing Problem, on which any of the five motion adaptive deinterlacers we have tested on it fails – as does any simpler deinterlacers. As seen in Figure 3.3 our variational motion compensated deinterlacer is able to solve The Interlacing Problem in this case.

Not all cases of The Interlacing Problem can be solved with our variational MC deinterlacer, as estimating the optical flow precisely on the perforated structure of interlaced image volumes is not a simple task, which the `Truck` example

[4]The Faroudja DCDi® is generally rated as the best deinterlacer (and video processor) there is in the world of high-end home entertainment, see e.g. `www.hometheaterhifi.com`.

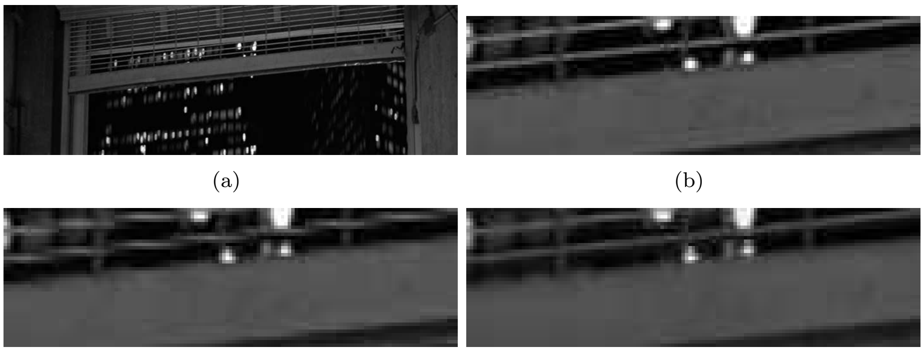


Figure 3.3: Part of the progressive sequence `Grille` in (a) with closeup zoom in (b). The security roller grille is moving downwards at a speed of ca. 2 pixels per frame ($\approx 1/4$ vertical grille space). This sequence is then artificially interlaced. (c) deinterlaced output using our variational motion adaptive algorithm (the four other motion adaptive deinterlacer tested on `Grille` give similar or worse results). (d) deinterlaced output of our variational motion compensated deinterlacer, the result being almost identical to the original with unbroken grille lines and sharp, straight edges.

in Fig. 3.2 illustrates (especially when viewed as video). Even with motion compensated deinterlacing increasing the density of information available for the actual deinterlacing, The Interlacing Problem embodies the ill-posedness of doing deinterlacing. We can however hope to get asymptotically close to the perfect solution of the problem. Using modelling of the human visual system and real world physical constraints on recorded image sequence content as in our suggested variational MC deinterlacing algorithm is a significant step in the right direction.

This chapter is organized as follows. In Section 3.2 we discuss deinterlacing in depth and give an overview of related work on deinterlacing including a number of known deinterlacers we have implemented and tested. In Section 3.3 we present our probability based variational framework and our two deinterlacing algorithms derived from it and before we conclude, we present our experimental work and deinterlacing results in Section 3.4.

3.2 Background and Related Work

In this section we will discuss other work on deinterlacing and present ten deinterlacing algorithms from literature, which we have implemented and tested along with our own two algorithms and the DCDi® deinterlacing in Faroudja's DVP-1010 video processor. We also discuss motion adaptive and motion compensated deinterlacing, optical flow estimation on interlaced video and the origin of variational deinterlacing.

3.2.1 Simple Deinterlacing Techniques

The simple algorithms given here can be found in video processing text books like [98] and [107] and in the 'must-read' book on deinterlacing by Bellers and de Haan, [4]. Starting with the spatial methods, Line Doubling (LDB) is the simplest interpolating the missing horizontal line by repetition of the above known line. LDB has been widely used in practice. Line Averaging (LAV), which is just a bit more advanced. LAV is the vertical fifty-fifty average of the above and below lines and is probably the most used deinterlacer in practice (often called 'bob'). In the comprehensive testing of deinterlacers in [4] and in our test as well LAV performs – given it simplicity – extremely well, which explains its popularity. The same cannot be said about the simplest temporal deinterlacer, Field Insertion (FI), a.k.a. merging or weaving. FI fills in the blank lines with neighboring lines in time and is basically a temporal version of LDB. FI was the algorithm used to simulate interlaced views of serration and line crawl in Figure 6.1(b) and its results are generally very similar to the images seen on an interlaced display, [4] and [107]. Field averaging (FAV) is a the temporal equivalent of LAV averaging the before and after temporal neighboring lines of each missing line. Vertical Temporal interpolation (VT) is a simple fifty-fifty combination of LAV and FAV. Many variations of vertical temporal filters have been suggested, e.g. by Thomas in [99]. All schemes mentioned so far are fixed, linear filters, whereas the algorithms we describe from here on are nonlinear and adapt to certain conditions in their local neighborhood and chose one of several possible interpolations depending on the local image content to yield better results. Or they are motion compensated.

Median filtering (Med) which is a real classic, is used for deinterlacing in many variations, see for instance [4], [16], [45], [94], [95] and [98]. For our testing we have chosen a 3-tap vertical spatiotemporal version from [4] although we use the forward temporal neighbor instead of the backward. In both cases the second and third taps are the two vertical spatial neighbors also used in LAV.

Spatial Edge Adaptive deinterlacing (EA) has been suggested in several forms, e.g. in [31], [63] and [98]. In all cases one tries to find dominating edge direction along which to interpolate with a skewed LAV filter.[5] This works relatively well on $\pm 45^o$ edges, but it fails miserably on data like the example in Figure 3.2. We have chosen to implement a scheme that determines the direction of interpolation by measuring the Summed Absolute Differences (SAD) along as set of candidate edge directions as described in [98]. We have modified it to detect the best of five directions, 0^o, $\pm 45^o$ and $\pm 63^o$ from vertical. We have now found seven of the ten deinterlacer we have implemented and tested (LDB, LAV, FI, FAV, VT, Med and EA), the remaining three are motion adaptive.

3.2.2 Motion Adaptive Deinterlacing

Motion adaptive deinterlacing (MA) can be done in a countless number of ways, but all MA deinterlacers have to do some form of per frame/region/pixel motion detection to switch off temporal interpolation when motion is present in a frame/region/pixel. Some graduate the switch-off depending on the amount

[5]The edge direction mentioned here is along the edge and not the more common crossing direction (gradient direction).

or strength of the motion detected. We have chosen to describe and implement three MA deinterlacers.

The first is suggested in [94] and [95]. It does explicit per pixel motion detection and takes advantage of the qualities of simpler schemes under different conditions: FAV when no motion is detected, spatiotemporal median filtering when the motion is detected to be slow and LAV when fast motion is detected. Thresholds classify the motion. We denote it MA1.

The second algorithm we have implemented is weighted vertical temporal deinterlacing (MA2) and is a simpler motion adaptive deinterlacer than MA1. MA2 uses a smooth weighted transition between temporal and vertical interpolation instead of a hard switching between schemes as in MA1. MA2 is described in detail in [63], where it is the second level in a successive approximation scheme. The idea of the successive approximation is to weigh the deinterlaced output of the current level with the result(s) from the previous one(s). The first level of approximation in [63] is LAV, the second the MA2 and the third is and edge adaptive scheme doing edge detection on the output of the MA2 deinterlacer.

In our tests we have implemented as a third motion adaptive deinterlacer (MA3) the third level of the successive approximation, but have used our own edge adaptive deinterlacer, EA from Section 3.2.1 above, which works directly on the interlaced input. We thereby take the successiveness out of the scheme but in also remove the possibility of error propagation. Error propagation is the big risk of successive methods where the faults caused by the simplicity of methods used first might no be robustly discovered and removed later. We will discuss the problem of error propagation in optical flow estimation on interlaced sequences in Section 3.2.5.

The video processor we have used to benchmark our own algorithms against is the Faroudja DVP-1010 video processor, which is considered the state of the art deinterlacer in the high-end home cinema market today. The deinterlacing of the Faroudja processor known as DCDi® (Direction Correlated De-interlacing) is basically a motion adaptive algorithm with a motion detector (MD) classifying the amount of motion on a 5-10 step scale. When there is no motion a median spatio-temporal filter is applied, which is then turned stepwise off as the amount of motion detected increases while a the output of a purely spatial filter is stepwise mixed in. The spatial filter, which takes completely over when the highest level of motion is reached is edge adaptive (a.k.a. direction correlated) detecting $\pm 45^\circ$ edges. If no dominant edge orientation can be detected, LAV is used as fall-back.

Another deinterlacing technology on the market today is Sony's X- Algorithm which is part of the signal processing unit sold with their top of the line professional broadcast/film HD LCD monitors (see `www.sonybiz.net/lmd` for further details). The X-algorithm is also motion adaptive and like Faroudja's DCDi® it uses a seven directional edge adaptive filter in pixels where motion is detected. We have not had the chance to test this algorithm, but sincerely doubt that the extra directions will improve results on e.g. the `Truck` sequence (Figure 3.2) as it is unlikely that the dominant edge direction detected will be the right one. The thin, brighter metal grid lines will most likely not be chosen over the darker background not even with very robust detection and switching, there is simply to little information available locally in each frame. Even with our five directional EA we doubt that interpolation in the $\pm 63^o$ direction will take place very often.

3.2.3 Motion Compensated Deinterlacing

In the intensity interpolation part motion compensated deinterlacing is basically the same as motion adaptive deinterlacing except on very important difference: One replaces the local temporal information at the same (x, y)-coordinates with $t \pm 1$ information along the flow trajectories. In theory this gives you access to correct temporal information in all pixels and high quality results are no longer obtained only in stationary parts of the frame. In practice there are some factors preventing you from getting perfect results in all cases. We will discuss the three most important of these factors.

First and least important, flows often point to spatially non-grid point in your discrete grid, thus invoking a need for spatial interpolation to get the $t \pm 1$ intensity values.

Second, optical flow methods do not guarantee precise and reliable flows in general wherefore we need local spatial interpolation – like in motion adaptive deinterlacers – as a fall-back option where temporal information is not sufficiently reliable.

Third, and most important, precise and reliable optical flow estimation is more difficult on interlaced video than on progressive video, and most existing optical flow methods are developed for use on progressive video. Using any of these methods on interlaced image sequences is not straightforward. The typical approach to interlaced optical flow estimation is using a simple non-motion compensated deinterlacer to create an initial progressive sequence to be used in the optical flow estimation, as discussed in [4] and done in [63]. This of course influences the accuracy of the resulting obtained flow due to the interlacing artifacts introduced by the simple deinterlacer used for initialization. Spatial interpolation is always the only reliable choice of any non-motion compensated deinterlacer in regions of motion, and thus the problem of obtaining an accurate flow will ironically be most severe in the regions where an accurate flow is needed the most: Regions troubled by The Interlacing Problem.

In [4] Bellers and de Haan thoroughly test 16 deinterlacers, among them 11 motion compensated methods, of which some are developed by the authors of [4]. The 11 MC deinterlacers use flows generated by block matching motion estimation algorithms also described in [4]. Using mean square error (MSE, to some known as the L_2-norm) as a quality measure when testing the 16 algorithms on a set of sequences with motion, the simple line averaging scores an average MSE of 43 and four of the motion compensated algorithms actually scores worse (45-72). The seven algorithms performing better than LAV scores 36-27 thus performing up to 37% better. Even when introducing two other measures, MSE_i and MTI (see [4] for details) that use the flow in calculating the error, it does not show that MC deinterlacing is significantly better than simple deinterlacing. It is a problem that no motion adaptive deinterlacers are included in the test. Since they are in output quality somewhere in between simple deinterlacing and MC deinterlacing, it widens the gap, it questions how good the MC deinterlacers in test are compared to MA deinterlacers. The argument of not considering motion adaptive deinterlacing is, that the motion detection is almost as complex as motion compensation – at least when using block matching. Block matching is when implemented right a very fast way to estimate optical flows, but due to the simplicity in modelling (translational motion only etc.) block matching flow are not among the most precise and reliable. That is, we believe, a common handicap of the 11 MC deinterlacers tested in [4]. Objective measures are however not the optimal method for evaluating deinterlacing results as it models very poorly how the human visual system judges the quality of deinterlacing. In [4] some subjective testing is presented, but not to an extent that provides full evidence of the superior performance of the presented motion compensated deinterlacers.

At least one of the MC deinterlacer from [4] have been used in circuits for Philips TV sets and progressive DVD-players.[6] Even though we have raised some critique here, the book [4] is highly recommended reading and it should be considered that the methods are at least seven years old (in 2007).

In [6] a newer example of a motion compensated deinterlacing algorithm is presented. Video versions of the deinterlacing results that could earlier be downloaded from the authors web site, did not really show the full potential of MC deinterlacing as e.g. the `Apollo` example still had severe serration artifacts.

There exists no common benchmark for performance evaluation for deinterlacing and there is an ongoing discussion on whether objective measures are really worthwhile, as they are not good models of subjective quality as judged by human observer (see Section 2.3 of this thesis). The human visual system, the perception, sensation and psyche of humans is simply to complex to easily model in objective error measures. The work by Biswas *et al.* in [5] is a resent attempt to come up with a performance analysis for MC deinterlacing, but [5] mostly focusses on the effect of errors in the flow vector field. Quality is measured using MSE on output deinterlaced sequences, but does not provide a generic, objective benchmark for deinterlacer performance.

We have chosen not to implement any motion compensated deinterlacers besides our own. Deinterlacers and optical flow algorithms are very complex and we wish to focus on our framework and algorithms derived from it. A discussion on the problems of benchmarking can be found in Section 5.2.8 later in this thesis.

3.2.4 Variational Deinterlacing

The idea of doing deinterlacing using variational methods was born when comparing the problems of deinterlacing and inpainting. Inpainting is to recreate regions in an image or film lost due to scratches, blotches etc. Deinterlacing is creating new lines in a video sequence, but considering the lines as having been lost during the recording of the scene, the problems can be considered similar.

A presentation of our variational framework for deinterlacing derived from an variational inpainting framework [65] is given in Section 3.3.1. From literature study we deduct that we are the first to use variational methods for deinterlacing. Variational methods for inpainting are still relatively new and since they are also computationally heavy, they do not seem immediately suited for use in deinterlacing, which mostly has to be done in realtime at the viewer end of a broadcasting or other video distribution chain (in a DVD-player, tv-set, set top box etc.). Inpainting can be done on a render farm over night (or longer) before printing a new and restored copy of the (old) damaged film, thus allowing the

[6]Both authors are or have been affiliated with Philips Research.

developer to prioritize quality without limitations on running times, memory requirements etc.

Variational methods for optical flow computations have been widely known since the pioneering work [48] by Horn and Schunck was published in 1981. The most accurate optical flow algorithm to date is presented in [9] and is also variational. Both these methods and most other variational methods can be integrated into both our variational framework for deinterlacing as well as in it inpainting ancestor in [65]. Generally variational optical flow algorithms produce the best and most accurate flow field of any methods available today, as the survey on motion estimation in [12] shows.

Both variational methods for inpainting and optical flow computations are computationally expensive. Recent work by Bruhn *et al.* in [11] shows that variational optical flow computations can be done in real-time on a standard PC. Since the flow computation is the heavy part of a variational motion compensated deinterlacer, variational MC deinterlacing in real-time seems realistic, especially as we aim at doing hardware implementations using field programmable gate-arrays (FPGAs). As variational motion adaptive deinterlacing is less complex than MC deinterlacing, real-time implementations are also within reach here.

We have so far only seen one other publication presenting work similar to ours in term of methodology. In [102] inpainting techniques are being used for deinterlacing, doing edge adaptive deinterlacing with edge orientations being detected by computation of structure tensors. (See [103] for details on structure tensor based inpainting.) The edge adaptive deinterlacing is successively followed by a motion compensated step, which do not do further deinterlacing, but do shutter-modelling to create motion blur and thus make video look more film like when combining the two types of footage in film/video production.

3.2.5 Optical Flow Estimation in Interlaced Video

As discussed earlier in Section 3.2.3 optical flow estimation on interlaced video is not straightforward. Known optical flow methods for progressive video need to be redesigned before they can be applied or the interlaced video has to be preprocessed in some fashion.

The very common approach of using a simple non-motion compensated deinterlacer to create an initial progressive sequence for optical flow computation (see e.g. [63]) is potentially troublesome. Any use of prior deinterlaced data in the optical flow computation might propagate errors caused e.g. by remaining interlacing artifacts, the problem being worst in areas with details in motion (i.e. The Interlacing Problem). Thus there is a risk of getting unreliable and unprecise optical flows and poor deinterlaced output quality.

Due to these risks we wanted to calculate the optical flow in another way, but we are faced with a hen-egg problem: In the new lines, should you compute the flow or the intensities first? In 'solving' the hen-egg problem we do in all cases need some optical flow and/or intensity initialization, which should be calculated under as few assumptions as possible to minimize error propagating from doing too simple preprocessing.

We have considered six possible solutions computing either a flow field to be used in the actual MC deinterlacing or in some way interleaving optical flow and intensity computations.[7] The six suggested solutions are listed here in order of estimated effect on visual output quality:

1. *The cost efficient approach*: Work at half the vertical resolution using the original pixels of the odd fields and the simple deinterlaced odd lines of the even fields to get odd line data for the full sequence. (Or only use the even lines instead.) This cuts the computational cost by 50% but then requires a vertical doubling of the flow field as postprocessing.

2. *The standard approach*: Do optical flow estimation on a simple deinterlaced version of the full progressive sequence.

3. *The 'half a pixel up half a pixel down' approach*: Use original data only and correct for the misalignment of half a pixel when vertically doubling the flow field. This seems a simple solution but in practice it will not be easy to handle top and bottom boundary conditions and all motions will be imposed a up-down zigzagging motion requiring some adaption. Alternatively one could use information at half-grid positions in every other field but this is nontrivial to do in actual implementations and would bring us close to solution 1) suggested above.

4. *The original data only approach*: Use given input data only but process the even fields and the odd fields separately as two independent progressive sequences. This is what we actually chose to use in our variational MC deinterlacer. To obtain the final flow field from the two separately computed even and odd 't to $t+2$' flow fields a 'fusion-scaling-smoothing'-scheme is needed.

5. *The idealistic approach*: Develop an interlaced optical flow estimator which is specifically designed to run on the perforated interlaced data volume. There is an unknown complexity to this task, e.g. in the obvious idea of allowing for flows to go through not yet deinterlaced lines and point at further away existing lines (flow vectors could have lengths one, two or more in time). To our knowledge these kind of methods have never been tried and furthermore they would require some post-interpolation of flows vectors for the new lines.

6. *The simultaneous approach*: Develop a motion compensated deinterlacer which simultaneously calculates intensities and optical flows in the new lines as finding the flow and intensities in the new lines where neither exists is really a 'hen-egg' problem. We have worked on this subject in video super resolution (Chapter 4 of thesis) and temporal super resolution (Chapter 5) but have not (yet) done any simultaneous deinterlacing. The basic idea is to alternate between optimizing flow and intensities and iteratively improve both as better and better versions of the other becomes available, or in its truest (an most likely only theoretically possible) form solve an integrated system optimizing both at the same time. Simultaneous variational deinterlacing will be discussed further in Sections 3.3.1 and 3.3.7.

[7]The term field is used for both interlaced frames and for fields of flow vectors. To avoid confusion we will try to use flow field whenever we talk of flow vectors of an image sequence frame and say fields (odd and even) whenever we talk of a frame of intensities missing every second line and thus being interlaced.

The idealistic approach is unlikely to be useable in practice and the simultaneous approach still to come. We strongly believe that using only original data followed by a good fusion-scaling-smoothing-scheme – solution 4 above – will give the best results of the four remaining solutions. The 'half a pixel up half a pixel down' approach seems almost as unrealistic as the idealistic approach and the cost efficient approach is just too bad. Doing variational optical flow on pre-deinterlaced video using the standard approach might yield usable results, but would compromise the quality by leaving artifacts or over-smoothing the output as smoothing the flow field would (partially) remove the effects of the artifacts but also result in a loss of details.

Several approaches to doing MC deinterlacing without using the standard approach and risking error propagation from that is discussed by Bellers and de Haan in [4] and some of these methods are likely to rank close to our 'original data only' approach (as far as the block matching methods used can produce optical flows of the same quality as variational methods) but the iterative nature of the simultaneous approach is the closest we come to solving the hen-egg problem of accurate and reliable computations of intensities and flows in deinterlacing, as it is the only way to overcome any possible problems with having to rely on simple and possibly erroneous initializations.

3.2.6 Deinterlacing in Colors

Good and computationally costly deinterlacers are normally only applied to the luminance/brightness channel of a color video signal with the chrominance channels only being simple deinterlaced. In [4] and [94] it is specifically mentioned as the method used (the simple deinterlacer for the color channels being line averaging), but other than that, exact strategies for color deinterlacing is not mentioned (in the approximately 50 deinterlacing papers and patents we have read).

This simple approach is in most case sufficient because the human visual system is much less sensitive to changes in color than to changes in luminance. This property of the HVS is used in both analog video and broadcasting and in digital video and broadcast to subsample the two color channels of the signal with at least a factor of two compared to the luminance channel with only very little or no loss in quality. In analog video and broadcasting it is done by halving the bandwidth of each of the two chrominance channels compared to the bandwidth of the luminance channel [70] and in digital media (using standards like MPEG-2 and MPEG-4/H.264) by simply subsampling the two color channels as thoroughly described in [107].

We did a few test to see if advanced deinterlacing of the chrominance channels improve quality but found line averaging to be more than adequate to give optimal color deinterlacing quality.

3.3 Theory and Algorithms

3.3.1 Variational Framework

The framework we present was originally proposed for image and image sequence inpainting, but as described in Section 2.6 it can be used in general for image sequence restoration and enhancement. It can be used as a framework for video super resolution – increased spatial resolution of each frame in the video – and temporal super resolution – changing the frame rate in a given sequence. Here we will focus on its use in deinterlacing.

The generic framework for image sequence inpainting and motion estimation (or motion recovery in the inpainting terminology) was proposed by Lauze and Nielsen in [65]. Starting with a Bayesian formulation of the problem we denote the observed damaged image sequence u_0 and define it to be on a spatiotemporal domain denoted Ω with D being the locus of the missing data in Ω, that is $D \subset \Omega$. $\vec{v}$ is the optical flow field of the restored and de-noised (optional) image sequence u, the desired outcome of the inpainting process. The joint probability of $\vec{v}$ and u knowing u_0 and D is factored as

$$p(u, \vec{v}|u_0, D) \propto \underbrace{p(u_0|u, D)}_{P_0} \underbrace{p(u_s)}_{P_1} \underbrace{p(u_t|u_s, \vec{v})}_{P_2} \underbrace{p(\vec{v})}_{P_3} \tag{3.1}$$

where u_s is the spatial distribution of intensities and u_t is the temporal distribution of intensities. P_0 is the likelihood for the image sequence, P_1 the spatial prior on image sequences, P_3 the prior on flow fields and P_2 acts as both a likelihood for the flow fields and a temporal prior on image sequences. The optimum solution, the maximum a posteriori (MAP) is then sought for in order to reconstruct the image sequence and recover the flow field. To transform the Bayesian formulation into a variational one we use Mumford's Bayesian to variational rationale from [75], $E(x) = -\log p(x)$. We are then given a continuous minimization problem of the form

$$E(u, \vec{v}) = E_0(u, u_0) + E_1(u_s) + E_2(u_s, u_t, \vec{v}) + E_3(\vec{v}). \tag{3.2}$$

Under mild regularity assumptions, a minimizing pair $(u, \vec{v})$ must satisfy the condition $\nabla E(u, \vec{v}) = 0$ where ∇ is the gradient and the solution expressed by the coupled system of equations

$$\begin{cases} \dfrac{\partial E}{\partial u}(u, \vec{v}) & = 0 \\ \\ \dfrac{\partial E}{\partial \vec{v}}(u, \vec{v}) & = 0. \end{cases} \tag{3.3}$$

The above problem formulation is used in [65] to develop and implement an image sequence inpainter that pseudo-simultaneously minimizes the energies for both the flow and the intensities in a multiresolution scheme – The necessary split in (3.3) is the step preventing perfect theoretical simultaneousness in the solution of the hen-egg problem. On each level of the pyramid first a the flow energy is minimized and then the energy of the intensities is minimized. The values are then warped to the next level, which has a bit higher resolution. The decrease in resolution from the finest to the coarsest level should be large enough to make the biggest hole to be inpainted small enough to get an optimally inpainted output. There is no analogy to this strategy in deinterlacing as scaling down by a factor of two vertically would remove all missing lines. The multiresolution approach is therefore only used for the flow optimization in deinterlacing, and we cannot obtain the same level of integration between flow and intensity calculations. In that sense the deinterlacing problem is simpler than inpainting problem – the holes are small already – but on the other hand 50% of the sequence is missing data, which would be considered difficult to recover if it were a case of inpainting. Luckily we know exactly where we have to create new data (detection and masking of scratches etc. is hard to automate and is often done by hand or semiautomatically in inpainting). Still the odd-even data distribution of the interlaced sampling grid is generally considered the curse of deinterlacing and it what makes deinterlacing a hard problem.

3.3.2 Selecting the Terms of the Energy

Selecting the model (some descriptive measure on the image sequence content and its distribution in real image sequences) to use for each of the E_i-terms in (3.2) is a tradeoff between mathematical tractability and maximum obtainable quality – models describing 'real' image sequence content well are mathematically complex. Here is a short discussion on how each term is selected.

$E_1(u_s)$ should punish too large local spatial variations in intensities and also try to model the true spatial intensity distributions in the new pixel positions. Using total variation on the spatial gradient in E_1 allows for smooth regions as well as edges.

The last term, $E_3(\vec{v})$, should enforce a reasonable local smoothness of the flow field. It is assumed that the displacement in the image sequence is caused by objects moving so that each pixel does not necessarily have an individual displacement but will have a displacement similar to that of (some of) its neighbors. Just as in $E_1(u_s)$ we need smooth regions – this time of the flows – with edges in the flow field at boundaries between objects moving differently, thus total variation is also the obvious choice for $E_3(\vec{v})$, but here applied to the local 3D spatiotemporal gradient of the flow field.

The term $E_2(u_s, u_t, \vec{v})$ states that the image content information should stay the same along the flow vectors and penalizing large discrepancies and is the most complex of the three terms. This term operates on both the intensities and the flow. It is a representation of the brightness constancy assumption or the optical flow constraint (OFC), see [9] and [48] and Section 2.5 of this thesis. Using total variation on $E_2(u_s, u_t, \vec{v})$ allows smooth flows and temporal edges, occurring at occlusions thus helping to solve the occlusion problem. To improve performance under changing brightness (e.g. from changes in lighting) $E_2(u_s, u_t, \vec{v})$ is in [9] and [65] extended to also include the gradient constancy assumption (GCA). Since both these terms operate badly on smooth regions (low spatial gradient of the intensities) the flow prior E_3 helps fill in motion vectors in these regions.

The term $E_0(u, u_0)$ in (3.2) is the well-known data or likelihood term from energy and probability formulations of diffusions. In the new pixel positions in deinterlacing it has no function, but in the known pixel positions it tells how much to diffuse, i.e. it controls the degree of diffusion imposed in de-noising.

3.3.3 Variational Motion Adaptive Deinterlacing

Looking at Figure 3.3 it seems a waist of time to do motion adaptive deinterlacing, especially when one's framework allows for motion compensated deinterlacing. But as we aim at hardware implementation at some point, we also need to carefully examine the potential of this more cost efficient form of deinterlacing. Also, it gave us a good starting point for doing variational deinterlacing and obtain a deeper understanding of the task before venturing into the more complex task of doing variational optical flow estimation and motion compensated deinterlacing. The algorithm we give here has also been presented as the best of two algorithms given in [55]. In [55] we did not introduce the overall variational framework and the focus was on whether to use a local 3D prior or to split the prior into a 2D spatial prior and a 1D temporal prior.

In order to get a variational motion adaptive deinterlacer from the presented framework, we simply set the flow to zero and thus only have to minimize

$$\frac{\partial E}{\partial u}(u, \vec{v} = 0) = \frac{\partial E(u)}{\partial u} = 0.$$

Since we assume $\vec{v} = 0$ the term E_3 in (3.2) disappears and the task of optical flow (motion) estimation is replaced by the need to do motion detection. Using the variational approach with total variation as the distribution chosen for the term E_2 in (3.2), which is now only a temporal prior on the intensities, we do not need explicit motion detection. Total variation in E_2 allows temporal edges and thus gives us implicit motion detection: When there is motion there is also a local temporal edge that total variation detects and will not diffuse across and thus we have implicit motion detection and adaptation.

The likelihood term on the intensities, E_0 in (3.2) is chosen to be a hard constraint as it simply leaves all original lines unchanged as it is customary in world of deinterlacing – but not the world of variational approaches. In e.g. de-noising, the data term[8] primarily acts as a *re*action to the diffusion action of the remaining terms controlling the degree of de-noising. With the data term being $u = u_0|_{\Omega/D}$ and plugging in the total variation terms, the general energy formulation can be rewritten to

$$E(u) = \int_{\Omega/D} (\alpha_s|\nabla u| + \alpha_t|\partial_t u|)\, dx, \qquad u = u_0|_{\Omega/D} \tag{3.4}$$

where ∇ is the spatial image gradient, ∂_t the local temporal gradient and the constants α_s and α_t the weights of the spatial and temporal priors. This is almost the energy to be minimized in total variation motion adaptive deinterlacing (denoted VMA), but since the function $|\cdot|$ is not differentiable at the origin, we replace it by the approximation $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$, with $\varepsilon = 0.1$ or 0.01 in our experiments and obtain

$$E(u) = \int_{\Omega/D} \left(\alpha_s\psi(|\nabla u|^2) + \alpha_t\psi(|\partial_t u|^2)\right) dx, \qquad u = u_0|_{\Omega/D}. \tag{3.5}$$

The spatial gradient term is thus a variation of the term introduced by Rudin et al. in [88]. In order to do the actual minimization, we us the fact, that $\psi'(s)/s = 1/\psi(s)$ and from using calculus of variation we obtain the Euler-Lagrange equation

$$-\alpha_s \mathrm{div}_2\left(\frac{\nabla u}{\psi(|\nabla u|^2)}\right) - \alpha_t\partial_t\left(\frac{\partial_t u}{\psi(|\partial_t u|^2)}\right) = 0, \qquad u = u_0|_{\Omega/D} \tag{3.6}$$

[8]Likelihood term and data term are aliases for the E_0-term.

where div_2 is the spatial divergence operator. We have chosen to solve this system using gradient descent and the gradient descent equation corresponding to (3.5) is

$$\partial_\tau u = \alpha_s \mathrm{div}_2 \left(\frac{\nabla u}{\psi(|\nabla u|^2)} \right) + \alpha_t \partial_t \left(\frac{\partial_t u}{\psi(|\partial_t u|^2)} \right), \quad u = u_0|_{\Omega/D} \tag{3.7}$$

where τ is the evolution time of this iterative solver. The numerics of the solution is given later in Section 3.3.5. In (3.6) and (3.7) it can be seen clearly that a large gradient either temporally or spatially will shut down the diffusion contribution of its own term, thus preserving edges spatially as well as temporally, the latter being the implicit motion adaption of VMA.

Additional Explicit Motion Detection

The setting of the weights α_s and α_t adds extra adaptivity to the implicit motion adaptivity of the algorithm. A very thorough test of different settings was done in [57], but is was found that optimal results on different test sequences required different weight settings and even internally in sequences (and frames) different weights would be required to get local optimal results.

Trying to improve results we added a local adaption for the weight of the temporal interpolation part. The spatial weight α_s was fixed at 1.0 while the temporal weight α_t was found by

$$\alpha_t = \frac{1}{1 + (C/C_0)^{2n}} \quad \text{or} \quad \alpha_t = 0.2 + \frac{0.8}{1 + (C/C_0)^{2n}} \tag{3.8}$$

where the first is basically a Butterworth filter and the second a Butterworth with the minimum value of w_t lifted from 0 to 0.2. C is the measure of how much motion is present at a location as found by a motion detector. Further details on (3.8) including testing along with a discussion on motion detection in general can be found in Section 3.4.5.

3.3.4 Variational Motion Compensated Deinterlacing

Dropping the assumption of zero motion and returning to the full motion compensated scheme using the full potential of the framework presented earlier in Section 3.3.1 and using the terms as described in Section 3.3.2, the energy (3.2) is instantiated as

$$\begin{aligned} E(u, \vec{v}) \quad = \quad & \lambda_0 \int_{\Omega \backslash D} (u - u_0)^2 dx + \lambda_1 \int_\Omega \psi(|\nabla u|^2) dx + \\ & \int_\Omega \psi\Big(\lambda_2 \left| \pounds_{\vec{V}} u \right|^2 + \gamma \left| \pounds_{\vec{V}} \nabla u \right|^2 \Big) dx + \\ & \lambda_3 \int_\Omega \big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2) \big) dx \end{aligned} \tag{3.9}$$

where ∇ is again the spatial gradient operator, ∇_3 is the local spatiotemporal gradient, λ_i and γ are some constants and v_1 and v_2 are the x- and y-components of the flow field, i.e. $\vec{v} = (v_1, v_2)^T$, and $\vec{V} = (\vec{v}^T, 1)^T$. The two $\pounds_{\vec{V}}$-terms are the $\vec{V}$-directional derivatives of u and ∇u respectively, also known as the Lie-derivatives (see for instance the book on Riemannian geometry by Gallot *et al.* [42]). To set the notation straight we have[9]

$$\pounds_{\vec{V}} u = \frac{\partial u}{\partial \vec{V}} = \nabla u \cdot \vec{v} + u_t = \vec{V}^T \nabla_3 u \approx u(\mathbf{x}, t) - u(\mathbf{x} + \vec{v}, t+1) \tag{3.10}$$

where the right hand side of the approximation is the brightness constancy assumption and the other terms are all notions for the linearized version of the brightness constancy assumption denoted the optical flow constraint (OFC). $\nabla_3 u$ is the spatiotemporal gradient of u. We also have that

$$\pounds_{\vec{V}} \nabla u = \frac{\partial \nabla u}{\partial \vec{V}} \approx \nabla u(\mathbf{x}, t) - \nabla u(\mathbf{x} + \vec{v}, t+1) \tag{3.11}$$

where the right hand side of the approximation is the gradient constancy assumption and the two other terms are its linearized version. The notation $\partial \cdot / \partial \vec{V}$ is used to some extent in literature on image and shape geometry but is problematic, as e.g. $\partial \vec{V} / \partial \vec{V} \neq 1$ although it would be considered equal to one in standard (partial) differential notations. Thus to avoid confusion we do not use this notation.

Returning to (3.9) ψ is the same as in the motion adaptive algorithm given in (3.5). As it can be seen in (3.9) diffusion is also allowed outside the region of missing/new data allowing for de-noising and of course imposing some smoothness as the price to pay for de-noising. Solving the system according to (3.3), the flow part loses the E_0- and E_1-terms and is thus a slight rewriting of the energy proposed by Brox *et al.* in [9] and is

$$E(\vec{v}) = \int_\Omega \psi\Big(\left|\pounds_{\vec{V}} u\right|^2 + \gamma \left|\pounds_{\vec{V}} \nabla u\right|^2 \Big) dx + \lambda_3 \int_\Omega \big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2)\big) dx \tag{3.12}$$

leaving out λ_2 for obvious reasons. The resulting Euler-Lagrange equation to be discretized – with the data term E_2 in vectorial form and the regularization term E_3 in scalar form for increased readability – is

$$\frac{\partial E}{\partial \vec{V}} =$$

$$2\psi'\left(\left|\pounds_{\vec{V}} u\right|^2 + \gamma \left|\pounds_{\vec{V}} \nabla u\right|^2 \right) \Big[(\pounds_{\vec{V}} u) \cdot \nabla u(x + \vec{V}) + \gamma \mathcal{H}\big(u(x + \vec{V})\big)(\pounds_{\vec{V}} \nabla u) \Big] = 0$$

$$\frac{\partial E}{\partial \vec{v_i}} = \lambda_3 \mathrm{div}_3 \left(\frac{\nabla_3 v_i}{\psi(|\nabla_3 v_i|^2)} \right) = 0, \qquad i = 1, 2 \tag{3.13}$$

where div_3 is the 3D local spatiotemporal divergence operator and $\mathcal{H}$ is the spatial hessian.

This is the equation we find an optimal solution to when minimizing the flow energy and is run on each of the separated even and odd fields (now progressive) in the first part of the 'the original data only approach' to finding the flow on an interlaced in progressive out sequence (see Section 3.2.5) We will return to the 'fusion-scaling-smoothing' part of the approach in Section 3.3.4.

[9] $\partial f(x)/\partial \vec{V} = \lim_{\epsilon \to 0} \;\; f(x + \epsilon \vec{V}) - f(x) \;\; /\epsilon = \nabla f(x) \cdot \vec{V}$.

For the intensity part of the calculations we have simplified things a bit compared to the energy given in (3.9) (which is used as is for inpainting by Lauze and Nielsen in [65]). The gradient constancy assumption part of the E_2-term has been skipped. It serves it purpose very well in the flow calculations to produce accurate and reliable flows under changing lighting conditions, but is very complex to work with when the Euler-Lagrange equation for the intensity energy $E(u)$ is derived as discussed in Section 2.6. On the negative side, we do not know if including the GCA would actually improve the results we obtain from our variational motion compensated deinterlacer, but inpainting results presented in [64] both with and without the gradient constancy assumption in $E(u)$ indicate that for small regions of missing data, the difference in output quality between the two is likely to be small.

The energy we minimize for the intensities when the forwards and backwards flows have been calculated is then

$$E(u) = \lambda_0 \int_{\Omega/D} (u - u_0)^2 dx + \lambda_1 \int_{\Omega} \psi(|\nabla u|^2) dx + \lambda_2 \int_{\Omega} \psi(\left|\pounds_{\vec{V}} u\right|^2) dx. \quad (3.14)$$

The Euler Lagrange equation derived from this energy still using $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is

$$\begin{aligned} \frac{\partial E}{\partial u} &= \lambda_0 \chi(u - u_0) - \lambda_1 \mathrm{div}_2\Big(\psi'(|\nabla u|^2)\nabla u\Big) \\ &\quad -\lambda_2 \mathrm{div}_3\Big(\psi'(|\pounds_{\vec{V}} u|^2)(\pounds_{\vec{V}} u)\vec{V}\Big) = 0 \end{aligned} \quad (3.15)$$

where χ is the characteristic function taking on the value 1 in Ω/D and 0 in D, and the discretization of the last term suggested by the approximation given in (3.10). λ_0 can either be set to zero to do pure deinterlacing without de-noising leaving all original pixels (u_0) untouched in the output similar to what almost all other deinterlacers do. $\lambda_0 = 1$ includes de-nosing (any other value of λ_0 is just similar to scaling λ_1 and λ_2 with the inverse of the same value). As argued by Bellers and de Haan in [4] de-noising can help remove some temporal flickering noise. We agree on this, but wish to emphasize the importance of carefully controlling the de-noising to avoid over-smoothing and loss of valuable information on details.

Algorithm Part I, Variational Optical Flow Estimation on Interlaced Video

We assumed to know the flow, when solving (3.15), but as discussed in Section 3.2.5 we cannot apply the variational optical flow algorithm that minimizes the energy in (3.12) directly on interlaced video. Here is our two step algorithm for flow calculations on interlaced video.

First we extract the odd field sequence, i.e. the sequence of the known lines for odd fields, and 'compress' it vertically by removing the unknown lines. In the same way, we produce the even field sequence (yielding a field distance of twice the input filed distance). The algorithm is:

1. Compute the flow $\vec{v}_o$ for the odd field sequence and the flow $\vec{v}_e$ for the even field sequence applying the implemented solution of (3.13).

2. Join $\vec{v}_o$ and $\vec{v}_e$ in order to produce a candidate flow field for the full progressive sequence using *fusion-scaling-smoothing.*

3. Repeat step 1 and 2 on the time-reversed sequences to calculate backward flows as well.

The first step we have already described, for the joining of the two flow fields $\vec{v}_o$ and $\vec{v}_e$ both containing flows from fields at time t to time $t+2$, we have to *fuse* – interlace them in time – then we have to *scale* twice, first the vertical component of the flow is double to accommodate the full frame height, the flow vectors are halved in length to be from time t to time $t+1$. Finally we have to *smooth* the new flow field to get the two parts to be consistent with each other and to fill in good estimates in the new empty lines of the sequence to be deinterlaced. This all ends up being *fusion-scaling-smoothing* and can be expressed mathematically as the energy

$$E(\vec{v}) = \lambda \int_{\Omega/D} (\vec{v} - \vec{v}_{o/e})^2 dx + \int_{\Omega} \Big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2) \Big) dx \qquad (3.16)$$

we can then minimize. Composing $\vec{v}_{o/e}$ in (3.16) is the fusion-scaling part, and the full minimization is the smoothing. ψ is the same as earlier. The fusion is done by just interleaving the two flow fields and the initial values in the empty lines are found by line averaging. The scaling is elementary as we assume linearity of the flow from time t to time $t+2$ and just halve the vectors. It is then up to the smoothing step to complete the process of producing one optimally merged and consistent flow field. For the Euler-Lagrange equation of (3.16), the first term is – after composing $\vec{v}_{o/e}$ – the same as E_0 in (3.14) just acting on the flow now. The second term is the same as E_3 in (3.12), which ensures coherence and give a feel of integration between the two steps of the full flow calculation. In our testing we skipped the de-noising of the already computed flow, that is we set $\lambda = 0$, as we had high confidence in the already computed flows. More about this in Section 3.4.

Algorithm Part II, Variational Motion Compensated Deinterlacing

Now we know how to obtain backwards and forwards flow, and all that remains to be done is using them in minimizing (3.15).

The algorithm for the intensity calculations is very similar to the one used for motion compensated sequence inpainting by Cocquerez and Chanas in [23] and by Lauze [64].[10] The optical flow algorithm we use is also fairly well-known from (see for instance [9] and [64]), so the part of the overall scheme predicted most likely to cause trouble is the full task of doing optical flow estimation on interlaced sequences including the fusion-scaling-smoothing algorithm. The scheme presented here has also been described very superficially in [53].

3.3.5 Numerics and Solvers

So far we have only presented Euler-Lagrange equations, that is partial differential equations (PDEs) in the continuous domain. We need implementations in the discrete domain to actually minimize the energies on digitally sampled image data: We need numerical solutions to our problems and have used different well-known methods of discretizing PDEs.

[10]In [64] inpainting both with and without the GCA in $E(u)$ are implemented and tested.

Motion Adaptive Deinterlacing

Our target is to solve the gradient descent equation in (3.7) and we do so explicitly, using forward difference for the evolution derivative ∂_τ and central difference for the divergence terms. (Approximations of continuous derivatives as discrete differences are described in several books, e.g. [89] and [1]). The forward difference of the evolution time derivative in (3.7) is a Taylor approximation of $\partial_\tau u = \partial u/\partial_\tau$:

$$\begin{aligned} \delta_\tau^+ u = \frac{\partial u}{\partial \tau} &= \frac{u(x,y,t;\tau+\Delta\tau) - u(x,y,t;\tau)}{\Delta\tau} \\ \Updownarrow \quad & \\ u(x,y,t;\tau+\Delta\tau) &= u(x,y,t;\tau) + \Delta\tau \cdot \frac{\partial u}{\partial \tau} \end{aligned} \tag{3.17}$$

Using (3.17) all we have to do is to add the time step $\Delta\tau$ multiplied with the output of (3.7) to the current value of each pixel. For the 2D spatial divergence term in (3.7) we have used three different schemes, one using a 4-point neighborhood of the current pixel and two using a full 8-point neighborhood, as described in [1]. These filters are very similar to those described below for the motion compensated deinterlacer. It is a well known (and proven) fact that using gradient descent with spatial filters of the type we use, the time step should be $\Delta\tau \leq 0.25$ in (3.17) and using any values larger in e.g. de-nosing or inpainting will give an unstable solution (typically the whole image will turn uniformly average grey). But in testing our variational MA deinterlacer, we found that even with $\Delta\tau \in [0.5 - 1.5]$ depending on the sequence our scheme stayed stable, owing to the fact that the filter support is mostly fixed pixel values from original, untouched lines.

Optical Flow

We have made our numerical implementation along the lines of [65] and [9] using a fixed point solver to isolate the linear part of the system, which is then solved using a Gauß-Seidel solver run until convergence or until a maximum number of iterations is reached. For the optical flow estimation it is still possible to use a multiresolution setting, which will handle large motions easily and speed up convergence. We thus use multiresolution as in [65] and [9], employing a down- and up-sampling scheme described in [11].

For the fusion-scaling-smoothing, after deriving an Euler-Lagrange equation of (3.16), discretization is done using standard methods. The first term is similar to the E_0 in (3.14) just operating on both flow components v_1 and v_2. The second term is the same as the E_3-term of (3.13) and thus treated in the same way.

Motion Compensated Deinterlacing

To solve (3.15) we use a Gauß-Seidel solver with a fixed point strategy to linearize the nonlinear system as described e.g. in Section 5.3.6 of this thesis. The discretization is as follows. We first look at the spatial term of (3.15) and following the ideas of discretization of total variation on the spatial 2D gradient given in [1], [18], [64] and [88] we can do a simple rewrite

$$\mathrm{div}_2\left(\frac{\nabla u}{\psi(|\nabla u|^2)}\right) = \mathrm{div}_{xy}(A\nabla u), \qquad A = \frac{1}{\psi(|\nabla u|^2)} \tag{3.18}$$

remembering that $\psi'(s)/s = 1/\psi(s)$. The spatial gradient is

$$\nabla u = (\partial u/\partial x, \partial u/\partial y)^T = (\partial_x u, \partial_y u)^T. \tag{3.19}$$

Now from the definition of the 2D divergence we have

$$\mathrm{div}_{xy}(A|\nabla u|) = \partial_x(A\partial_x u) + \partial_y(A\partial_y u). \tag{3.20}$$

Using (3.19) and (3.20), equation (3.18) can be approximated finitely as ([1])

$$\mathrm{div}_{xy}(A\nabla u) \approx \delta^o_{x,h/2}(A\delta^o_{x,h/2}u) + \delta^o_{y,h/2}(A\delta^o_{y,h/2}u) \tag{3.21}$$

where $\delta^o_{x,h/2}$ is the central difference approximation of $\partial u/\partial x$ and $\delta^o_{y,h/2}$ the same in the y-direction. h is the distance between two horizontally or vertically neighboring pixel positions (grid points) and most often set to 1. By definition

$$\begin{aligned} \delta^o_{x,h/2} &= \frac{(u(x+h/2,y,t) - u(x-h/2,y,t)}{h} \\ \delta^o_{y,h/2} &= \frac{(u(x,y+h/2,t) - u(x,y-h/2,t)}{h}. \end{aligned} \tag{3.22}$$

Before we do the next rewrite we switch to compass coordinates where the indexes w, e, n, s represents the directions in the grid and is short for west, east, north and south, $u_w = u(x-1, y, t)$ and so on. u_c is the center (or current) pixel. Plugging (3.22) in (3.21) and rewriting gets us the divergence approximation

$$\begin{aligned} \mathrm{div}_{xy}(A\nabla u) \approx\ & \frac{1}{h^2}\Big((A_{w/2}\cdot u_w + A_{e/2}\cdot u_e + A_{n/2}\cdot u_n + A_{s/2}\cdot u_s) - \\ & (A_{w/2} + A_{e/2} + A_{n/2} + A_{s/2})\cdot u_c\Big). \end{aligned} \tag{3.23}$$

We need the A's at half-grid point, and we get them by calculating A's at all grid points and then interpolating them at the half-grid points. We could use linear interpolation, i.e.

$$A_{w/2} \approx (A_c + A_w)/2 \tag{3.24}$$

but

$$A_w \approx \frac{2}{\dfrac{1}{A_c} + \dfrac{1}{A'_w}} \tag{3.25}$$

endorses edge preservation. Say $A_c = 100$ (flat region) and $A'_w \approx 0$ (edge), then the approximation would be $A_w = 50$ from (3.24) corresponding to a flat region and $A_w \approx 0$ from (3.25) corresponding to the close by edge we wish to preserve. To calculate the A's at grid points, the choice of $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is clever. Since $1/|\nabla u| = 1/\sqrt{(\partial_x u)^2 + (\partial_y u)^2}$ the choice leads to

$$A \approx \frac{1}{\sqrt{\frac{1}{4}\left((u_e - u_w)^2 + (u_n - u_s)^2\right) + \varepsilon^2}} \tag{3.26}$$

when using central differences and is similar to the 2D total variation term used for inpainting by Chan and Shen in [18]. This is also the 4-point neighborhood scheme used for the motion adaptive deinterlacer. The first of its two 8-point scheme also uses the diagonal directions, and the second weighs the 4-point diagonal with the 4-point standard given in (3.26) according to a detection of edge directions. Extensive testing of the performance of the three in [57] showed practically no difference in output quality, and thus we chose to use the 4-point standard scheme in (3.26) only, as it approximately a factor of two faster than the two 8-point schemes.

For the temporal part we have a 1D divergence, div, or simply a derivative along the flow. Since the flow is subpixel precise, the intensities at the position the flow points to is mostly not at grid point and has to be interpolated. We use bilinear interpolation for that. u_c is once again the current, center pixel, $u_a = u(x + v_x, y + v_y, t + 1)$ is the after intensity value along the forward flow and $u_b = u(x + v_x, y + v_y, t - 1)$ is the before intensity value along the backward flow and we have

$$B_a \approx \frac{1}{\sqrt{(u_a - u_c)^2 + \varepsilon^2}} \quad \text{and} \quad B_b \approx \frac{1}{\sqrt{(u_c - u_b)^2 + \varepsilon^2}}$$

This lead us to the complete direct solution

$$u_c = \frac{\lambda_1(A_w \cdot u_w + A_e \cdot u_e + A_n \cdot u_n + A_s \cdot u_s) + \lambda_2(B_a \cdot u_a + B_b \cdot u_b)}{\lambda_1(A_e + A_w + A_n + A_s) + \lambda_2(B_a + B_b)} \tag{3.27}$$

where the A's and the B's are the nonlinear components. In case we include the data term, the full solution becomes

$$u_c = \frac{\lambda_1(A_w \cdot u_w + A_e \cdot u_e + A_n \cdot u_n + A_s \cdot u_s) + \lambda_2(B_a \cdot u_a + B_b \cdot u_b) + \chi \cdot u_{0,c}}{\lambda_1(A_e + A_w + A_n + A_s) + \lambda_2(B_a + B_b) + \chi} \tag{3.28}$$

where $u_{0,c}$ is the current pixel in the input (χ is one for known lines and zero for new lines). We run a number of outer fixed point iterations where we update the A's and the B's once in each followed by a number of inner relaxation iterations (the Gauss-Seidel solver) where we update u_c by solving (3.27) or (3.28).

3.3.6 Iterative Schemes

A basic advantage of using iterative schemes in combination with a temporal aperture spanning both backwards and forwards in time is an increased spatial and temporal coherence when calculating either intensities or flow fields. By iterating information from further away than the ± 1 neighbors is propagated to the current pixel. The use of total variation ensures that only relevant information is propagated as it stops diffusion over edges. In the intensity calculations e.g. in our variational deinterlacer, knowing the flow (i.e. motion compensation) of course increases the temporal information propagation.

3.3.7 Simultaneousness

With the above given motion compensated deinterlacing algorithm we lose the simultaneousness that its ancestor the MC inpainting had since it no longer makes any sense to include the intensity part in the multiresolution pyramid.

A way of getting close to the simultaneous approach suggested in Section 3.2.5 is to have a reasonably precise initialization of a) the flow for all pixels and b) the intensities in the empty lines. Then one would alternate between updating intensities and flow in each iteration. This would in practice be as simultaneous as the MC inpainter and is the same as what we do for MC video super resolution in Chapter 4.

For deinterlacing it might not prove as worthwhile to increase the simultaneousness as it clearly will in inpainting and temporal super resolution (see Chapter 5, which both have large unknown regions. The results presented in the next section support this claim as they show that we are close to an optimal solution of the interlacing problem.

3.4 Experiments

In this section we present the results obtained with our variational deinterlacers and benchmark them against other deinterlacers. Before getting to the actual results some general information about our test setup is given.

3.4.1 Quality Measurements

We use both objective and subjective quality evaluation the deinterlacers in test, but give focus to the subjective evaluation. The example given in Figure 3.3 of the sequence `Grille` illustrates why: In spite of the subjectively large difference in quality between variational motion adaptive and variational motion compensated deinterlacing on `Grille` the objective difference measured using the mean square error (MSE) is insignificant (and in favor of the MA deinterlacer!) as only a small percentage of each frame is affected by the problems in the grille. The MSE and most other objective measures are global and have problems giving significant weight to local problems very disturbing to the human observer. Objective measure like the MSE could be used more locally but then the output MSEs becomes a large data set itself and the evaluation of it tedious and possible also subjective (when sorting and selecting what to give as results). The goal of our evaluation is to measure how pleased the human visual system is with our deinterlacing results and thus subjective evaluation is the best measure, which is also acknowledged in industry, e.g. at the consumer electronics producer Bang & Olufsen [100].

The exact form of the MSE used is

$$\text{MSE} = \frac{1}{N}\sum_{D}(u - u_{gt})^2 \tag{3.29}$$

where u_{gt} is the ground truth and we sum over the missing lines D only, making N equal to half the number of pixels in the sequence. The MSE cannot be used on image sequences that only exist as interlaced video, but only on artificial interlaced video, which is produced by removing every other line of a progressive sequence, the progressive sequence providing the ground truth. Some measures enabling objective evaluation of deinterlacing of true interlaced material are given in [4] and have been discussed in Section 3.2.3 of this chapter, but they rely heavily on the optical flow computed for a sequence. Further discussion on objective vs. subjective evaluation can be found in [4] and Section 2.3 of this thesis (and references therein).

3.4.2 Test Material and Online Video Results

In testing we used both true interlaced video and interlaced material made from removing lines in progressive video. The truly interlaced material was recorded using a Sony VX-1000 DV camera while some truly interlace and all progressive origin material has been taken from PAL DVD's. We have chosen sequences challenging the deinterlacers and several sequence are plagued by the interlacing problem. All tests were conducted on the 8 bit luminance channel.[11] Online video results of running variational motion compensated deinterlacing on the test sequences `Building`, `Copenhagen Pan`, `Truck` and `Credits` are given at `http://image.diku.dk/sunebio/DI/DI.zip`.

3.4.3 Initialization Tests

For all variational deinterlacing results given in this paper we initialized the new lines of the input sequences by line averaging. This is to give our iterative variational deinterlacers a rough but reasonable starting point. To thoroughly test the influence of initialization we ran several test on our variational deinterlacers using other initializations. Initializing all new pixels with the median value of 128 we still ended up at the same results as when initializing with line averaging (LAV). The only difference was that convergence to the end result was faster from LAV. Using the output of the motion adaptive deinterlacer MA1 (described earlier in this chapter) as initialization did again give the same result as starting from LAV initialization, but with a marginally speedup in convergence. Our variational deinterlacers seems rather robust against bad initialization, but with Gaussian noise in the span [0-255] as the initialization we did not get convergence and were left with a rather noisy 'deinterlaced' image sequence.

3.4.4 Objective vs. Subjective Results

The ten deinterlacers mentioned as implemented in Section 3.2 and our variational motion adaptive deinterlacer has been thoroughly tested in [57] and part of this test is presented in [55]. We have included parts of this test along with some tests of these 11 deinterlacer on other sequences here to clearly show the advantages of both variational methods as such and variational motion compensated deinterlacing in particular.

An extensive test of the parameter settings was presented in [57] the conclusion being that tuning α_s and α_t in (3.7) can improve performance on a given sequence slightly, but we recommend setting $\alpha_s = \alpha_t = 1$, which makes the motion detection completely implicit.

Objective results for a number of sequences are given in Table 3.1. The MSE's for the two first test sequences, `Person` and `C&T` (from [55]), show that our algorithm is superior to the others in test. For the sequence `C&T` the MSE ranking is the same as the subjective ranking found by careful and repeated visual

[11] As mentioned in Section 3.2.6 we did conduct some tests on YC_rC_b color video, but are not further discussed here.

Scheme	`Person`	`C&T`	`Grille`	`Building`	`Credits`
Line Doubling (LDB)	17.90	79.72	151.7	92.8	516.7
Line Averaging (LAV)	5.53	26.31	63.7	*27.4*	159.8
Field Insertion (FI)	22.26	472.25	247.2	414.6	1639.0
Field Averaging (FAV)	9.03	284.22	162.4	253.7	908.5
Vertical Temporal (VT)	5.62	94.34	99.0	98.5	405.6
Median (Med)	9.27	65.72	127.8	89.2	509.8
Motion Adapt. 1 (MA1)	5.53	28.18	*63.1*	27.9	158.9
Motion Adapt. 2 (MA2)	5.07	67.97	110.5	83.3	337.5
Edge Adaptive (EA)	8.77	32.87	87.1	58.5	511.2
Motion Adapt. 3 (MA3)	5.36	48.79	95.7	61.2	342.5
Variational MA (VMA)	*4.97*	*26.06*	70.2	35.8	165.7
Variational MC (VMC)	–	–	80.3	44.7	*103.7*

Table 3.1: Objective results. The measures given here are mean square errors (MSE). The non-variational methods are described in Sections 3.2.1 and 3.2.2.

inspections of the deinterlaced outputs. But the small difference in MSE to the nearest competitors does not justify how much better our algorithm actually is when subjectively comparing the results. On `Person` our algorithm even produces a result hardly distinguishable from the original progressive version of the sequence. Only FAV and FI fails miserably on `Person` whereas the remaining deinterlacers give reasonable to good results, but none of them performs as well as our variational MA deinterlacer.

On `Grille` LDB is subjectively the worst, the remaining methods are not quite as bad with our VMA being the least bad of all. None of these 11 deinterlacers come close to the quality obtained with variational motion compensated deinterlacing as shown in Figure 3.3 although the MSEs in Table 3.1 rank it only forth. The ordering by subjective quality on `Grille` is not the same as the corresponding ordering by MSE's.

On `Building` the ordering by MSE is again different from the subjective ranking. The objective (MSE) top five is 1. LAV, 2. MA1, 3. variational MA, 4. variational MC, 5. EA. Subjectively the top five becomes: 1. variational MC, 2. EA, 3. variational MA, 4. LAV, 5. MA1. Even though VMC and EA scores significantly worse MSEs than the three deinterlacers in front of them, they are subjectively evaluated a lot better than all other deinterlacers in test as it should be clear from inspecting Figure 3.4. Besides the good subjective result on `Building` EA deinterlacing in general performs horribly as it mostly creates serious salt-pepper like artifacts.

On `Credits` variational MC deinterlacing is subjectively a lot better than any other deinterlacer in the test and although it is ranked first by MSE as well, the difference sown to number two does not justify the difference. Variational MA is ranked fourth by MSE but subjectively a clear number two and LDB which is tenth by MSE is subjectively fifth (after LAV and MA1 on a tied third place). VMA and VMC results are shown in Figure 3.5.

From the results given here and in [57] we see that MSE is not a good measure of deinterlacing quality – at least not on our test data. We will now

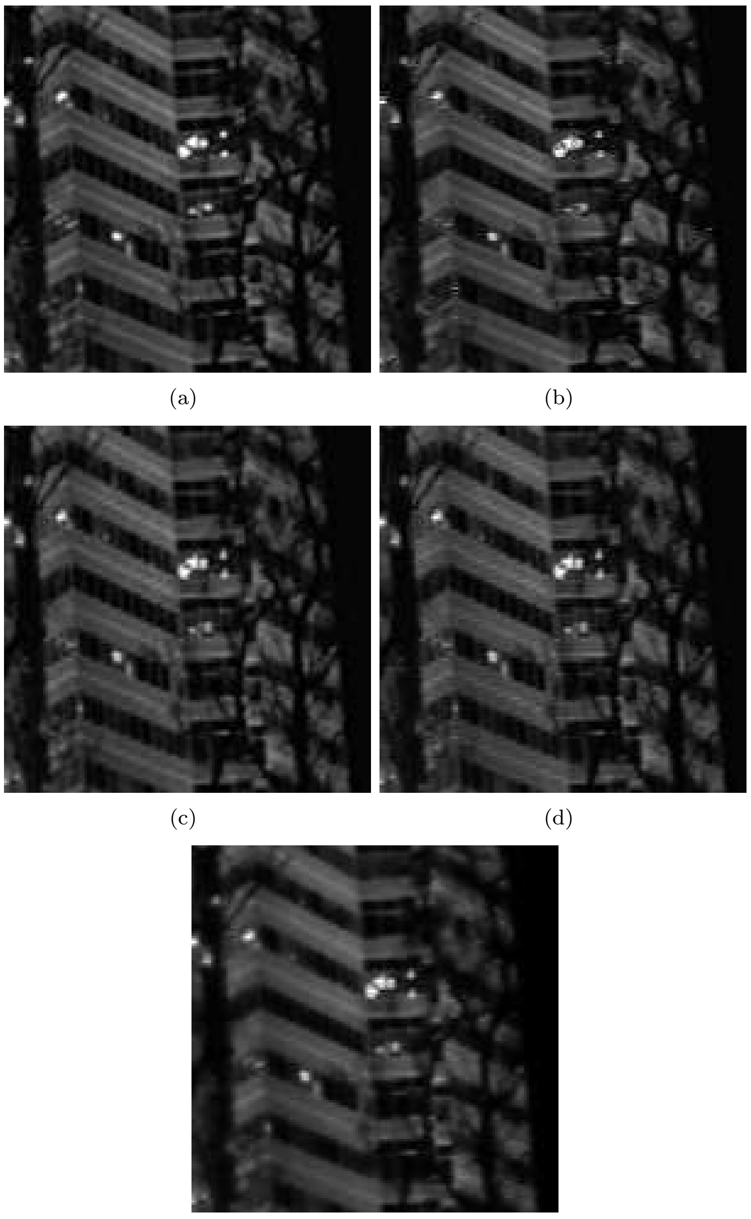


Figure 3.4: The sequence `Building` with tilting camera motion. (a) Original progressive, deinterlaced with (b) EA , (c) LAV, (d) variational MA, and (e) variational MC, which is very close to the original (a bit smoother due to denoising). EA is also close to the original but has some salt-pepper artifacts that are very noticeable during playback.

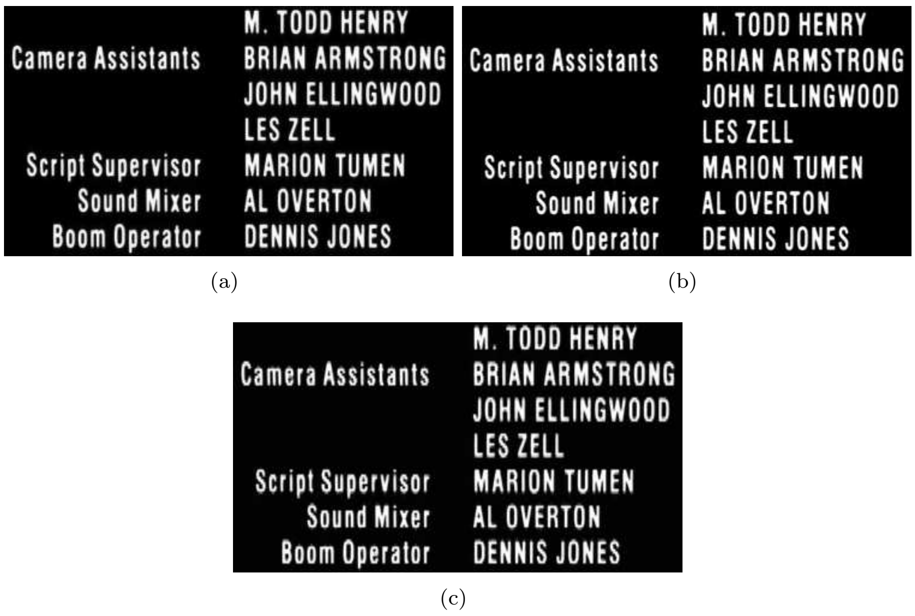

M. TODD HENRY
Camera Assistants BRIAN ARMSTRONG
JOHN ELLINGWOOD
LES ZELL
Script Supervisor MARION TUMEN
Sound Mixer AL OVERTON
Boom Operator DENNIS JONES
(a)
(b)
(c)

Figure 3.5: The sequence `Credits` has rolling end titles and the output of our variational MC deinterlacer in (a) is practically indistinguishable from the progressive original in (b) when viewed as video but appears a bit smoother as still (due to de-noising). The result of variational MA deinterlacing shown in (c) is also the second best performing deinterlacer on `Credits` and it does look quite good as still but has severe flickering when played.

determine which motion adaptive deinterlacer performs the best in terms of output quality.

3.4.5 Best Motion Adaptive Deinterlacer

In Figure 3.6 of the sequence `Copenhagen Pan`, which is true interlaced so that no MSE can be measured, it is clearly seen how our variational MA deinterlacer outperforms two other motion adaptive methods shown, MA1 and MA2. In our earlier tests presented in [57] and [55] MA1 and MA2 are the two methods that come closest to our variational MA deinterlacer in both subjective and objective quality, but they are still significantly worse. The bad performance of MA3 is most likely caused by our implementation of edge adaption (part of MA3, see Section 3.2.2) which is not as robust as could be. The Faroudja DVP-1010 video processor with DCDI® prides itself of being very good at edge adaptation and thus should be the best motion and edge adaptive deinterlacer there is. It is a 'closed box' system and due to Macrovision and HDCP copy protections we have not been able to do any objective evaluation of its deinterlacing performance and all subjective evaluation has to be done realtime. We conducted a A/B real time comparison test which was repeated on different screens (high quality plasma, LCD and DLP (projector) displays.) The DVP-101 came quite close to variational VMA in terms of subjective quality, but is not quite as good as VMA on average.

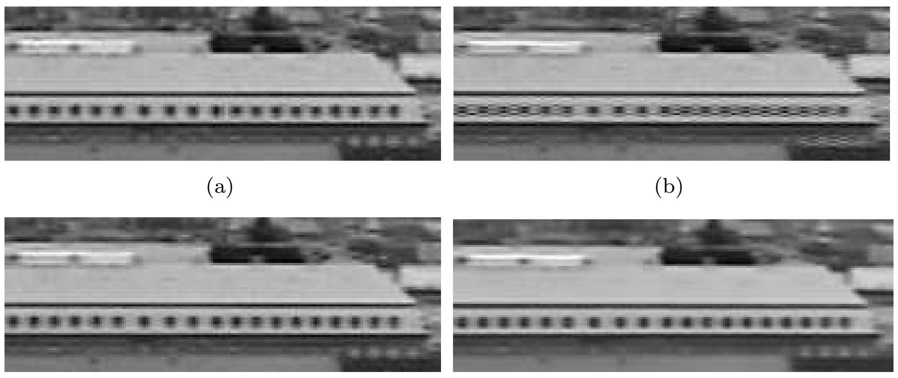


Figure 3.6: Cutout of `Copenhagen Pan` deinterlaced with (a) MA1, (b) MA2, (c) our variational MA, and (d) our variational MC. The motion in `Copenhagen Pan` is a ca. 5 pixel/frame camera pan to the left. MA1 gives median 'sparks' around the windows, MA2 serrates heavily, variational MA is almost perfect and variational MC just a bit better, mainly due to de-noising.

The result of VMA in Figure 3.6 is good, but still we do not show the full frame of `Copenhagen Pan`, because there are areas where all MA deinterlacers performs poorly. We have already seen on three examples that variational motion compensated deinterlacing is subjectively much better than any other method in test, but before we go into details of VMC deinterlacing test results, we will discuss the results of trying to improve our variational MA deinterlacer.

Additional Explicit Motion Detection

As described in Section 3.3.3 we aimed at improving our variational MA deinterlacer by using additional explicit motion detection. From extensive testing we found the optimal setting to be $C_0 = 6$ and $2n = 4$ using the second filter in (3.8). C in (3.8) is the measure of how much motion is present at a location. We used $C = |u(x, y, t-1) - u(x, y, t+1)|$ but also tested $C = |u(x, y, t-1) - u(x, y, t)|$, which proved unstable. In some cases the explicit motion detection falsely detected no motion in pixels where there was motion. This resulted in some weak serration artifacts making the overall performance of VMA a bit worse with explicit motion detection. Still the performance of our VMA was always marginally better than or equal to that of the Faroudja DVP-1010. But the general limitations in performance of MA deinterlacers – the inability to solve The Interlacing Problem – is the major reason why (variational) motion compensated deinterlacing is so interesting in spite of its increased complexity.

3.4.6 Variational Motion Compensated Deinterlacing Results

As seen from the results given so far, variational motion compensated deinterlacing performs much better than any motion adaptive deinterlacer, mainly because it solves The Interlacing Problem by properly deinterlacing highly detailed regions in motion.

	fixed point iterations	relaxation iterations	weights
Eq. (3.13) Flow estimation	5	20	$\gamma = 100$ $\lambda_3 = 70$
Eq. (3.16) Fusion-scaling-smoothing	10	100	$\lambda = 1$
Eq. (3.15) Deinterlacing	5	50	$\lambda_0 = 1$ $\lambda_1 = 0.1$ $\lambda_2 = 5$

Table 3.2: Recommended settings for variational motion compensated deinterlacing.

Parameter Testing and Intermediate Flow Results

There are more parameters to tune in the VMC than in the VMA deinterlacer. In Table 3.2 an overview is given of what we found to be the optimal setting after extensive testing. In all three parts a convergence threshold of 10^{-7} or 10^{-6} is set but never reached in testing.

For the motion estimation step we used an 80 level multiresolution pyramid with a scale factor of 1.04 between the levels, which we found to be optimal for our PAL resolution test material. Lowering the number of levels does affect the flow mildly, but any differences disappear almost completely when the flow is further processed with the fusion-scaling-smoothing. We suggest setting $\lambda_3 = 70$ as lower values will over-segment the flow: Smooth changes in flows at deformations will be segmented in separate parts and regions of uniform motion (e.g the hair of a person) will not be merged to a unform flow field but have a mishmash of incorrect flows (outliers) as seen in Figure 3.7(a). Higher values of λ_3 will provide a unnecessary smoothing of the flow as some regularization inevitable occurs in the fusion-scaling-smoothing step. Setting $\gamma = 100$ is a good choice. Higher values will possibly provide additional details to the flow, but these details as well as any over-segmentation are smoothed out in the fusion-scaling-smoothing step.

In the fusion-scaling-smoothing $\lambda = 1$ to 3 is appropriate: Higher values will produce a smoother flow, which can both help get rid of over-segmentation but also remove fine details in the flow. We ran 10 outer fixed point and 100 inner iterations with a convergence threshold of 10^{-6}. In Figure 3.7 it can be seen how typical flow outputs look before and after fusion-scaling-smoothing.

In the deinterlacing step optimizing the solution of (3.15), setting $\lambda_0 = 0$ and thus shutting off de-noising in original pixels results in more temporal flicker in moving detailed regions. The temporal weight λ_2 should be at least one, but values of five and 10 has also yielded good results, although they do introduce some smoothing in moving regions. We recommend and have used $\lambda_2 = 5$ with spatial weighting $\lambda_1 = 0.1$ in the results given in this section. The chosen setting of λ_1 minimizes temporal flicker in the output.

Generally our three step algorithm has – within limits – proven to be quite robust to changes in parameter settings, but further parameter testing might still improve performance. Especially the flow calculations can give very detailed (or controlled poorly, over-segmented) flows, but they are smoothed out

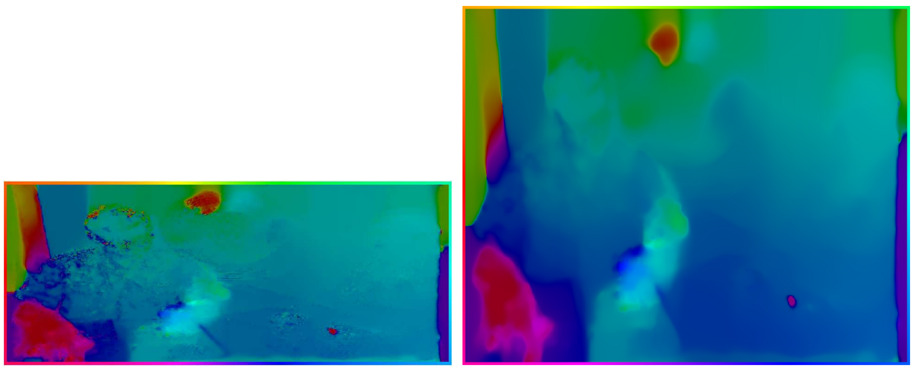


Figure 3.7: Typical flows, here from a sequence showing a person at a desk. (a) shows how the flow is typically over-segmented before fusion-scaling-smoothing. (b) shows how the fusion-scaling-smoothing not only upscales the flow, but also homogenizes it, removing incorrect flows (the over-segmentation) in the process. The border shows the flow directions coded by the hue, while the intensity codes the magnitude of the flow normalized in [0.5,1].

by the fusion-scaling-smoothing. It would be optimal to make sure the most accurate and detailed flow possible reaches the deinterlacing. Flow estimations on interlaced video is as discussed earlier in this chapter a hard problem and work on improved optical flow estimation for deinterlacing will most likely be the source of the biggest improvements of (variational) MC deinterlacing.

Running Times

We have not focussed on optimizing our code for speed (yet). The code is written in C++ using the CImg library (`http://cimg.sourceforge.net/`) interfacing with Matlab. The running times given here was found on a Windows XP PC with a 1.6 GHz Pentium M processor using the settings given in Table 3.2. On six frames of the sequence `Truck` (436×714 progressive spatial resolution) it typically takes just under 4 min. to compute each of the four initial flows (forward/backward, even/odd fields only). The fusion-scaling smoothing process takes approximately 45 sec. for each of the two flows (forward and backward). The intensity energy minimization takes approximately 3 min. 40 sec. In total that is 20 min. 40 sec. with the flow calculations taking up 84% of the total computation time. With 50 fps interlaced video, we are about a factor 1500 from realtime for the intensity part. For the slower part, the flow calculations, we know that simpler variational flow algorithms run in real-time on standard PCs (although on web camera resolution video) [11]. The running time results given are representative for all our tests.

Results

Three experts and two laymen have judged the output quality of our variational motion compensated deinterlacer in an A/B comparison with the Faroudja DVP-1010 video processor with DCDi®. On seven different test sequences all

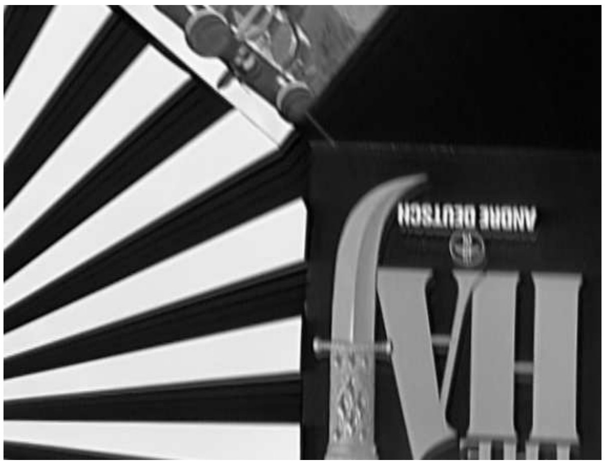


Figure 3.8: Variational motion compensated deinterlacing of the sequence `BBC3` (cutout shown). A few artifacts are seen in the still, but they are not observed during playback. The motion in `BBC3` is rotation around an axis in the lower left corner of the frame.

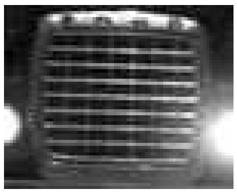
suffering from the The Interlacing Problem – detailed regions in motion – our algorithm clearly outperformed the Faroudja DCDi® according to all five test persons. The output from our variational MC deinterlacer on each of the three test sequences having a progressive original was judged almost indistinguishable from their original (again in an A/B comparison). Due to copy protection (Macrovision and HDCP) we are unable to produce stills or record video of the output from the Faroudja processor. We were however able to stream our own progressive results through the DVP-1010 to ensure that we were comparing the two deinterlacers directly.

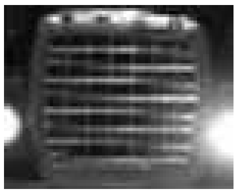
On the sequence `BBC3` (Figure 3.8) details are recovered and seen clearly and motion is smooth with the variational MC deinterlacer, whereas the Faroudja has staggering motion and the details are hard to see.

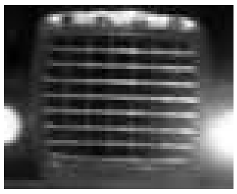
On the sequence `Truck`, which was used to illustrate the interlacing problem in Figure 3.2, we are able to (re)create the correct structure of the front grille whereas all other deinterlacer we have tested, MA and simple, fails at this and creates some dark, diagonal lines across the front grille. Comparing the result from our MC deinterlacer with the progressive ground truth in Figure 3.9 shows hardly any differences.

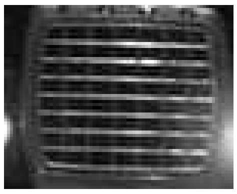
The superior performance of variational MC deinterlacing on `Grille` has already been discussed and is clearly seen in Figure 3.3. Figure 3.4 shows how variational MC deinterlacing also produces the best result on `Building`, although slightly smoothed it is clearly better than the second best, EA deinterlacing, as observed during playback. The sequence `Credits` in Figure 3.5 has rolling titles, a type of content causing most deinterlacers to perform very bad, but the output of our variational MC deinterlacer is practically indistinguishable from the progressive original during playback.

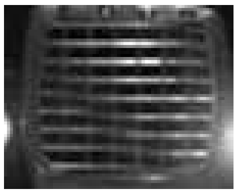
On the `Copenhagen Pan` cutout in Figure 3.6 the variational MC deinterlacer is not much better than variational MA except for a bit of de-noising. But all the MA deinterlacers we have tested including the Faroudja produce serious


(a) Ground truth


(b) Variational MA


(c) Variational MC


(d) Ground truth


(e) Variational MA

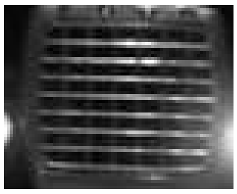
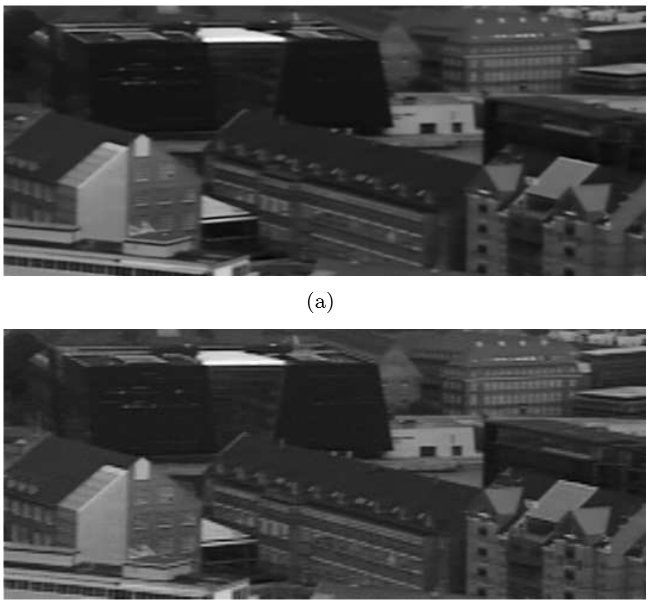

(f) Variational MC

Figure 3.9: Deinterlacing of `Truck` shown in Figure 3.2. (a)-(c) frame 5, (d)-(f) frame 17. The results of variational motion compensated deinterlacing in (c) and (f) is almost indistinguishable from the progressive ground truth in (a) and (d) while the best of the rest, variational motion adaptive deinterlacing in (b) and (e), creates some horrible artifacts.

flicker on other buildings in the sequence. The Faroudja also produces quite a lot of serration on the roof windows of the castle (the cutout in Figure 3.6) thus performing worse than variational MA. The variational MC deinterlaced output has no flicker and is overall perfect (although there is no progressive original to compare to) and Figures 3.10(a) and 3.10(b) shows how much better it is on another part of `Copenhagen Pan` compared to the best MA deinterlacer, our VMA.

So far we have presented only good results from our variational motion compensated deinterlacer, but we also encountered one bad result. On the sequence `Keyboard` with very complex motion (close-up of fingers typing fast on a keyboard and hand held camera) the motion was not recovered correctly and results in serration clearly observed when inspecting the frames as stills. Of the five persons in the panel only one (expert) observed the serration in `Keyboard`. Thus variational MC deinterlacing can be improved as no optical flow estimation algorithms is perfect. The serration in `Keyboard` also shows that we could improve robustness against unreliable flows in our variational MC deinterlacer, but improved flows calculations on interlaced video would probably help even more (robustness fixes a problem whereas better flows lifts the quality). The temporal coherence gained by calculating flows on sequences as one and not frame pair by frame pair is a major strength, but on dynamic motion with large accelerations it becomes a weakness. The problem is general for variational motion compensated video upscaling and we will analyze this problem further and suggest solutions in Chapter 5.

Unfortunately we do not have an easy way of comparing the quality of our deinterlacing results with results in literature. There is no common set of data and quality measure like the `Yosemite` sequence and the angular error measure used in the optical flow community. Comparing our results with the ones from



(a)

(b)

Figure 3.10: Another cutout of `Copenhagen Pan` (same frame as in Figure 3.6) showing the superiority of variational MC deinterlacing (a) over variational MA deinterlacing in (b) as it manages to solve The Interlacing Problem. Notice the stair step effect on almost horizontal and diagonal lines in the MA output, which also causes severe flickering during playback.

the survey by Bellers and de Haan in [4], there is too few subjective results given in [4] to judge by. Taking the relatively small gap in objective evaluated performance compared to line averaging and the fact that no motion adaptive schemes were tested, we believe to outperform all the schemes given based on the results of variational MC deinterlacing given in this section. It might be that one or more of the MC deinterlacers given in [4] could compete with our method, if the block matching was improved or replaced with a better motion estimation scheme. The methods in [4], however, have one major advantage over our variational MC deinterlacer: They all run in realtime.

3.5 Conclusion and Future Work

We have shown that using variational methods for deinterlacing is advantageous. The variational MA is best among simple and motion adaptive deinterlacers just outperforming the Faroudja DVP-1010 video processor with DCDi® which is considered state of the art in home cinema deinterlacing. But no MA deinterlacer including our variational MA deinterlacing solves The Interlacing Problem of detailed image regions in motion. This problem is solved by our variational motion compensated deinterlacer, which yields high quality results close to the ground truth. To fully prove its quality a benchmarking against other MC deinterlacers is needed. Our variational MC deinterlacer still has flaws mostly related to bad optical flow estimation, partially due to the problems of working on interlaced video. Improvements are to be expected if the flow and intensity calculations are integrated in a simultaneous scheme. In terms of running time, we are far from realtime in our current software version, but the realtime computations of variational optical flows presented by Bruhn *et al.* in [11] shows that realtime variational MA and MC deinterlacing is close at hand. Especially if dedicated hardware and parallelization are employed. Our algorithms repeating the same operations on all pixel over and making it an obvious target of parallelization.

Chapter 4

Video Super Resolution

Producing high image quality on high definition (HD) displays when showing standard definition (SD) material is a problem of upscaling frame resolutions from low resolution (LR) to high resolution (HR) and is as such a super resolution (SR) problem, or as we prefer: *Video super resolution* (VSR). In technology available today the problem is typically solved using simple spatial interpolation. Using motion compensated (MC) methods instead will allow for information transport along the optical flow trajectories of the video and increase the level of detail and sharpness in the high resolution output. We present a variational motion compensated VSR method that simultaneously computes the desired high resolution video and a high resolution flow fields to increase accuracy of the temporal information transport. Creating super resolution flows has to our knowledge not been done before. Most advanced SR methods found in literature cannot be applied to general video with arbitrary scene content and/or arbitrary optical flows as it is possible with our simultaneous VSR method, which also allows for arbitrary discrete magnification factors. We show in test that our variational simultaneous VSR algorithm outperforms other SR methods applicable to our general video problem, and we also attempt to break the limits of super resolution [2] in our experiments by increasing the frame resolution eight times in both height and width (8x8 VSR).

4.1 Introduction

Super resolution (SR) is a thoroughly investigated subject in image processing where a majority of the work is focussed on the creation of one high resolution (HR) still image from n low resolution (LR) images (see for instance [19]). In this chapter we will join the minority doing video super resolution (VSR) defined to be the creation of an n frames high resolution video from an n frame low resolution video. An overview of work done on VSR can be found in [38] although under the name multiframe super resolution.

Our motivation for doing VSR is to solve the problem of showing low resolution, typically standard definition (SD) video signals on high definition (HD) displays at high quality. The SD signals that are either broadcast or stored on DVDs, hard disks, etc. and could be in PAL, NTSC or SECAM and need to be upscaled before they can be displayed on modern displays such as LCD

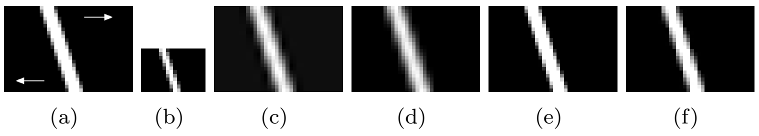


Figure 4.1: Frame no. 3 of 5 in the `Skew` sequence: (a) ground truth (arrows show skew motion), (b) ground truth downsampled by 0.5x0.5 with loss of information. 2x2 super resolution on (b): (c) bicubic interpolation, (d) bilinear interpolation, (e) our motion compensated VSR algorithm and (f) our algorithm run purely spatial.

and plasma tv-sets and most projectors (DLP, LCD etc.). Until high definition television (HDTV) takes over from current SD standards there will be a need for upscaling and even far into the future both broadcasters and private homes will have large archives of SD material that needs to be upscaled before being displayed. The increase in spatial resolution does not need to stop with the current HDTV formats, and in coding the availability of high quality upscaling will allow for lower bit rates and/or higher image quality.

But since millions of HD displays are already in use, upscalers are also in use. The problem is that they do not do a very good job as it is shown in Figure 4.1. Bilinear interpolation (for instance used in the expensive, high end, stand-alone video processors by Faroudja) and bicubic interpolations are the two very widely used upscalers and in Figure 4.1(c) it can be seen that bicubic comes a little closer to the ground truth (Figure 4.1(a)) than bilinear interpolation does (Figure 4.1(d)). Our motion compensated video super resolution using total variation filters is indistinguishable from the ground truth as can be seen in Figure 4.1(e).

The two standard methods bilinear and bicubic interpolation hardly qualifies as super resolution methods as they are simple spatial interpolation methods and by Borman and Stevenson's classification in [8] they are ranked lowest in the SR hierarchy. Depriving our method of its flow part and running it purely spatial on single frames, it still performs clearly better than the two standard methods as seen in Figure 4.1(f). From the examples in Figure 4.1 it should be clear that our method has the potential to improve the experience of SD video on HD displays.

The `Skew` example in Figure 4.1 is a simplification of the world and recordings of it, we cannot always expect to come this close to the ground truth. Super resolution is an ill-posed problem: One wants to reconstruct a hypothetical higher resolution image (sequence) from an image (sequence) sub-sampling at a lower resolution. The inevitable loss of high frequency information as foretold by the Nyquist-Shannon sampling theorem, causes the super resolution problem to be ill-posed.

The idea of doing super resolution (one HR image from multiple LR images) in spite of the limitations posed by the subsampling is coupled to how detail enhancement happens in the human visual system (HVS). The input resolution on the retina is not as high as the resolution perceived by the HVS after it has processed the input. The eye constantly makes small, rapid movements thus supplying the visual system with a number of low resolution views from which the HVS can construct a higher resolution, more detailed view. It has been debated for a long time whether the small, rapid eye movements did more then just stop the visual input from fading [87], but it was not proven until recently by Rucci *et al.* [87].

The suspected super resolution effect of the HVS might have inspired early work on super resolution in the early 1980s, e.g. the work by Tsai and Huang in [101]. It seems obvious that the unsteadiness of a video camera will give subpixel differences from frame to frame. From this spatial low resolution but temporally abundant data one should be able to mimic the HVS in creating higher resolution outputs. No matter if inspiration came from vision, signal processing, or both, it is the basic idea of super resolution that one can increase the level of details somehow inferring knowledge over a number of time and/or space shifted low resolution samples of a scene.

Ideally we wish to model what the HVS in computer vision also when it comes to doing super resolution. We build our model of video super resolution based on knowledge of how the lower level vision works. If one applies the right modelling in doing (video) super resolution, one should be able to generate high resolution images or image sequences that will please the HVS, or at least not disturb it with annoying artifacts.

In typical super resolution producing just one HR image, the n LR views are registered to a common frame of reference before the HR output is generated. The registration is often simplified by using different known transformations and subsamplings of the same image thus simplifying the registration. If the LR input is an n-frame video sequence, the registration is typically done by computing the optical flow from each frame to the common frame of reference.

In VSR we could do as in single frame SR by define a support area of e.g. ± 4 LR frames for each HR frame and slide a window along the sequence to produce n HR frames from n LR frames. This would be extremely heavy in terms of computational requirements and thus we will 'just' compute the standard backward and forward optical flows of our n frame LR sequence and use only the optical flow between neighboring frames in time to produce our n HR frames. We then claim that through solving our system iteratively, we will propagate information from more than just the two neighboring frames to each HR frame. We also benefit from having increasingly more and more detailed HR frames and not just LR frames to draw information from: We produce the n HR frames simultaneously. Furthermore we simultaneously update the optical flows of the sequence also in high resolution, which aids accurate detail propagation from frame to frame. Last but not least we use total variation as our main model spatially, temporally and spatiotemporally. Total variation allows for sharp edges, also temporal edges, and used within the right framework it does not introduce any artifacts as often seen with super resolution algorithms: Very often the prize payed for sharp edges is ringing artifact (see for instance several of the examples given in [47], [2] and [38]). By controlling the diffusion of our total variation model properly we avoid getting cartoon look of total variation run too extensively where the image is divided into very smooth, almost flat regions with sharp edges as boundaries.

This chapter is organized as follows. In Section 4.2 we present the details of the super resolution problem and look at other work on SR and VSR algorithms, and in Section 4.3 we discuss the theory and modelling behind our variational motion compensated VSR method before we give the actual algorithm. Finally we present our experiments and results in Section 4.4.

4.2 Background

4.2.1 The Super Resolution Problem

Before describing our algorithm or discussing related work on super resolution, we will present the basic model used in super resolution. In practice SR is a subsampling problem in the discrete domain as we have images or frames that are sample at a resolution lower than what is desired. As a subsampling problem it is an ill-posed problem, and even though it is a problem existing in the discrete domain it is possible to model it in the continuous domain. We follow [69] and start the analysis of the SR problem with the continuous image formation equation that models the projection $\mathcal{R}$ of a high resolution image into a low resolution one:

$$u_0(y) = \mathcal{R}(u)(y) + e(y) := \int B(x, y)u(x)dx + e(y). \tag{4.1}$$

$u_0(y)$ in (4.1) is the LR irradiance field, $u(x)$ the HR irradiance field, $e(y)$ some noise and $B(x, y)$ is a blurring kernel. In general this kernel is assumed to be shift invariant and takes the form of a point spread function (PSF). Replacing images with image sequences, x is replaced by (x, t) where t is the time dimension of the sequence.

The general discrete formulation is

$$L = RH + E$$

where $H \in \mathbb{R}^N$, $L \in \mathbb{R}^n$ with $n < N$ and L and H being the discretizations of u_0 and u respectively. R is a linear map $\mathbb{R}^N \to \mathbb{R}^n$ and $E \in \mathbb{R}^n$ is a random vector. Inverting this equation is severely ill-posed and more information is needed to get a stable solution. First, more than one low resolution view is used for a given HR view, which is the fundamental idea of SR and what happens in the HVS as described in Section 4.1. Secondly, some regularization of the problem is introduced, e.g. in the form of a prior on the distribution of HR images, $p(u)$. A common type of prior imposes spatial or spatiotemporal regularity. Another one (that can be used together with the first type of priors) relies on expected content of images. The general process can be described as follows: different LR images $L_1, \ldots, L_n$ are registered toward a common one, say L_1, via sub-pixel displacement fields $w_2, \ldots, w_n$ such that $L_1(y) = L_i(w_i(y))$, hoping to get as close as possible to the ground truth H.

Related to SR is image interpolation, which is used e.g. in image zooming where only one view of a given scene is available. One HR image is then created from just one LR image and one then has to rely completely on the regularization to supply information for the upscaling.

A crucial part of an SR algorithm is the modelling of the point spread function, but before we go into details with that, we will look at related work on (video) super resolution.

4.2.2 Related Work on Super Resolution

Pioneering super resolution Tsai and Huang in [101] modelled their approach of solving the super resolution problem in the frequency domain but did not take noise into account in their models. Kim *et al.* extended the formulation of Tsai and Huang to include noise in [60]. A different family of methods emerged directly in the spatial domain where a maximum likelihood estimator (ML) would produce HR output $\bar{H}$ which minimized the projection distances to all the sample LR images L_i. Adding priors to these approaches will replace the ML estimator by a regularized ML or a maximum a posteriori (MAP) like estimator as (see e.g. the work by Schultz and Stevenson in [90]).

An extensive review of different approaches to solving the super resolution problem is given by Borman and Stevenson in [8]. Another extensive bibliography can be found in [19], and an overview including some of the most recent work is found in [93]. In the work of Irani and Peleg [49] motion compensation (MC) is used to extract and register several frames from a given sequence and then create an HR image from them. Schultz and Stevenson [90] also integrate motion compensation as well as prior smoothness constraints more permissive for edges than the Gaussian one. Here too, authors aim at reconstructing one HR frame from a video sequence input. Generalizations of these methods have been used for spatiotemporal super resolution of video sequences by Shechtman *et al.* in [92]: From a set of low resolution sequences one high resolution sequence is generated (multi-camera approach).

The recent video super resolution algorithm by Farsiu *et al.* [38] models only affine or similar parametric optical flow and is a typical example of how modelling of the registration simplified. This is typical example of ignoring the complexity of the registration task in it self. Any super resolution algorithm applied on real world data[1] is no good without a reliable and precise registration, in the case of video the registrations becomes the complex task of computing optical flow.

In [37] Farsiu *et al.* introduce an interesting combination of demosaicing and video super resolution. As amateur video cameras only have one charge-coupled device (CCD) they can only sample one color per pixel and thus demosaicing is essential for obtaining full color video sampling (e.g. RGB). Professional cameras have 3 CCDs, one for each of the RGB channels, thus making demosaicing unnecessary.

Super resolution has its limits as shown by Baker and Kanade in [2] and recently by Lin and Shum in [69]. The limits is imposed by the fact that noise amplified by the super resolution algorithm will grow quadratically with the magnification factor making large magnifications (more than two on real image data) impractical. In order to overcome these limitations, Baker and Kanade have proposed doing hallucination, which is to add a generative, trained model part to the reconstruction. We want to be able to process any kind of video content and just as the limitations in the modelling of the registration in the algorithm of [38] (affine and parametric flow only) prevents us from using that method, we cannot use the method of Baker and Kanade either as it is optimized on a subset of video and image data only. Even the semi-generic, learned priors of Freeman *et al.* in [40] are too limited, as the patches used for training comes from images similar to the images used in test. We thus need to develop a generic VSR algorithm able to handle arbitrary scene content and optical flows without prior training.

[1]Typical test data is transformed (shifted/rotated/skewed) and then subsampled versions of one HR image. Thus the registration is just using the known transform.

4.2.3 The Foundation of Our Method and Related Work on Variational Methods

The starting point of our work is Bayesian inference. We derive a variational formulation and replace the optimization via maximum a posteriori (MAP) with an energy minimization whose optimization takes the form of a non linear reaction-diffusion spatially within each frame and temporally along the flow field. The optical flows composing the flow fields are themselves computed simultaneously with the intensities using variational methods in an integrated framework. This is a technique known from image sequence inpainting [65]. An earlier variational approach for motion compensated inpainting was presented in [23] computing first the flow and then the intensities as it is also done in variational motion compensated deinterlacing in Chapter 3 of this thesis. In our first version of variational motion compensated video super resolution in [54] we also used this sequential approach. The simultaneous approach for variational MC VSR also computing high resolution flows is presented for the first time here (a detailed description will be given in Section 4.3).

The use of a highly precise variational optical flow algorithms as part of our framework enables us to capture flows even more complex and detailed than the one in the skew sequence example in Figure 4.1, whereas even the skew motion would be a source of problems to the widely used block matching motion estimators.

Lillholm *et al.* [68] have presented a method for the reconstruction of images from a finite set of features measured on the image (linear filter responses) under smoothness constraints in a variational framework, a problem very similar to super resolution. In both cases the iterative minimization of the given energy will not be restrained to the correct solution hyperplane. The correct energy minimization can be found by orthogonally projecting the suggested energy minimization back onto the solution hyperplane. In the super resolution case the projection is dictated by the super resolution constraint $\mathcal{R} \cdot u - u_0 = 0$ that follows from the model of the image formation process given in Section 4.2.1 (assuming $e(y) = 0$ in (4.1)). In our Bayesian formulation, which will be given in Section 4.3, $\mathcal{R} \cdot u - u_0 = 0$ poses as the likelihood (data fidelity) term.

Our Bayesian framework models both super resolution on the sequence and the optical flow field formation and from that we derive a maximum a posteriori (MAP) approach simultaneously computing HR flows and HR image sequences. Shen *et al.* [93] presents a MAP approach that jointly computes a HR image, the flow field of the LR sequence it comes from, and a (flow based) object segmentation in one iterative, cyclic scheme. The segmentation, the flow from each LR frame to the reference frame and the HR image are considered interdependent, just as we would like to consider the flow of a sequence and sequence as integrated and as dependent on each other as possible. The model in [93] only allows for perspective, parametric motion, the spatial prior is the Laplacian (that smooths across edges) and the number of moving objects needs to be known in advance, which makes it unsuitable for our purposes.

Simultaneous registration and single HR image construction was also done earlier by Hardie *et al.* in [46] but without considering multiple motions in the scene.

Since we cannot use the advanced learned spatial priors of e.g. Baker and Kanade [2] we have chosen to use total variation, although more advanced generic regularization models are available, e.g. structure tensor based methods, which are used for image interpolation in [103] and [86]. Tschumperlé and Deriche does energy minimization without back projection as dictated by the image formation process in (4.1). Therefore Roussos and Maragos, who do do back projection get better results when comparing their method to the method of Tschumperlé and Deriche in [86]. The major disadvantage of structure tensor based regularization is the computational cost: Recalculating the structure tensor in each iteration is costly, but does give better results than just doing spatial total variation based regularization (as shown e.g. in [86]). Having temporal coherent frames available in an sequence of images (enabling us to do video super resolution instead of image interpolation) gives a potential of detail enhancement not possible in image enhancement. But the increased ability to produce sharp edges with structure tensors compared to total variation makes it interesting for future improvements.

Total variation is nonlinear and thus more complex to use than the linear Gaussian distribution but it will preserve and strengthen edges and is thus worthwhile using. The spatial bilateral total variation filter is suggested for VSR in [38] and shown to perform better de-noising than standard total variation on an artificial example (noisy text), but no comparisons of the two used on natural images are given in [38], neither for de-noising nor for VSR.

So far we have discussed scientific work on super resolution focussing on getting as close a possible to the ground truth. In actual products for video processing like the Faroudja and DVDO video processors, and in build-in processors in high-end DVD-players and displays (plasmas, projectors etc.) focus is on visual quality as judged by the human observers. In these devices the majority of the resources are typically spent on deinterlacing, noise filtering, correction of MPEG-2 errors and color corrections. Unfortunately bilinear interpolation as in Figure 4.1(d) is the standard method used for video super resolution as it is cheap, easy to implement and does not create severe artifacts. The smoothing produced is not and artifact severely unpleasing to the HVS, but given the trend of larger and better displays it will not suffice over time, sharper video is needed.

4.2.4 Modelling the Point Spread Function

When recording a far away star in astronomy it is crucial to get as much information from just one point sample as possible, and something similar can be said about recordings in microscopy. In the irradiance image formation model in (4.1) it is the same point spread function we are looking for. We need to model how the light is dispersed through the camera lens and sampled on the recording medium, the sensor. We have

$$PSF = PSF_{lens} * PSF_{sensor} \tag{4.2}$$

where $*$ is the convolution operator. To keep the modelling balanced between correctness and mathematical tractability, Gaussian or uniform (mean) distributions are typically chosen for the two terms. Lenses in cameras are of high quality and blurring usually not a problem. Blurring might be a problem in lenses for astronomic telescopes and microscopes and probably that is why the lens PSF has found its way into PSF modelling in super resolution. The problem with digital camera lenses is not blurring but more the opposite, that is aliasing. As it is pointed out e.g. in [4] anti-aliasing (low pass filtering) in lenses is very difficult in practice and when reading tests of digital still cameras the problem of moiré often comes up even with 10 megapixel cameras. Thus lens blur modelling has no place in PSF modelling in SR except for in very special cases.

Choosing between Gaussian and uniform distributions for the sensor PSF, the uniform distribution is the obvious choice for digitally scanned film recordings (not modelling the film granularity). The Gaussian mainly seems to be used when lens blur is included in the PSF model (e.g. in [37] and in [92]) maybe to fit a Gaussian downsampling of test data often used prior to running the SR algorithm.[2] A problem with the Gaussian is that it has one degree of freedom, the variance, which needs to be set but there is nothing in the model that suggest a certain value, thus it has to be found empirically for a given data set, and retuning for other data sets might be necessary. The uniform sampling is fixed, it is the distribution that most truthfully models the sampling across CCDs [3] and it is for instance used in [2], [69], [79] and [90] – and we will use it as well: Most digital video have been sampled using CCDs, either when recorded with digital cameras or scanned from film using modern telecines (film scanners).[3]

Roussos and Maragos [86] use the uniform distribution convolved with a Gaussian of large standard deviation, which they claim helps remove blockiness (a.k.a. jaggedness or 'jaggies'). The restricted use of Gaussians might also remove some moiré and noise, but one will of course risk removing fine details.

Temporal integration or point spread in time to model the temporal aperture of the recording is typically left out of the modelling. In [102] it is used to add film-like motion blur to video recordings when the two types of material is edited together (no (V)SR done). Both Patti *et al.* [79] and Lin and Shum [69] assumes the PSF to be uniform in time also, but Lin and Shum point out it should be time integrated to remove motion blur, this is however complex as one ideally should know the shutter time use in the recording. We also assume temporally uniform PSF and our model allows for any motion blur to be (almost) untouched as it is most likely a desired artistic effect of the film maker(s).

4.3 Variational Video Super Resolution

In this section we will go through the different aspects of designing and developing our algorithm for simultaneous computation of high resolution image sequences and their optical flows. Some aspects have already been mentioned briefly, for instance our starting point, a Bayesian framework, from which we derive our algorithm.

4.3.1 Bayesian Framework and Motion Compensated Image Sequence Upscaling and Restoration

The framework we present here was first formulated by Lauze and Nielsen in [65] to be used for simultaneous image sequence inpainting and motion recovery,

[2]Downsampling is done to enable a comparison between the SR results and ground truth.
[3]Telecines use 3 CCDs to scan in full RGB, so demosaicing is not needed for digitally scanned films.

we have used it for deinterlacing in Chapter 3 of this thesis, for temporal super resolution (frame rate conversion) in Chapter 5 and for a simpler version of video super resolution in [54]. Thus it spans widely.

We wish to model the image sequence content and its optical flow using probability distributions. The locus of missing data given in [65], is a necessary tool to model missing lines in deinterlacing and missing frames in temporal super resolution, but we have no regions of missing data in video super resolution, so we are left with

$$p(u, \vec{v}|u_0) \propto \underbrace{p(u_0|u)}_{P_0} \underbrace{p(u_s)}_{P_1} \underbrace{p(u_t|u_s, \vec{v})}_{P_2} \underbrace{p(\vec{v})}_{P_3}. \tag{4.3}$$

where $\vec{v}$ is the optical flow of the high resolution output sequence u, and u_0 is the low resolution input sequence. u_s and u_t are the spatial and temporal distribution of intensities respectively. On the left hand side we have the posterior distribution which we wish to maximize doing MAP. The right hand side terms are: P_0, the image sequence likelihood, P_1 the spatial prior on image sequences, P_3 the prior on motion fields and P_2 a term that acts both as likelihood term for the motion field and as spatiotemporal prior on the image sequence. The term spatiotemporal do not denote the spatial plane and the purely orthogonal temporal component, which is a commonly used, stringent definition of spatiotemporal in a 3D sense. We consider an image sequence to be 2D + 1D, time can not be juxtaposed with the third spatial dimension as step sizes cannot be said to be the same in time and space. More importantly, with motion in the sequence the relevant information is found along the motion trajectories, thus we define spatiotemporal as the 2D spatial neighborhood in combination with the information rich time dimension located along the optical flow field.

In the sequel we assume that u_0 comes from a noiseless image formation equation, i.e. $e(y) = 0$ in (4.1).

4.3.2 From MAP to Variational Energy Minimization

We use the Bayesian to variational rationale by Mumford [75], $E(x) = -\log p(x)$ to get to a variational formulation of our problem and replace MAP with an energy minimization. This means that the optimal pair $(u, \vec{v})$ minimizes the constrained problem

$$\begin{cases} E(u, \vec{v}) = E_1(u_s) + E_2(u_s, u_t, \vec{v}) + E_3(\vec{v}) \\ \mathcal{R}u = u_0 \end{cases} \tag{4.4}$$

where $E_i = -\log P_i$ and $\mathcal{R}$ the projection described in Section 4.2.1. The likelihood term P_0 is the constraint $\mathcal{R}u = u_0$ governing the mapping back and forth between high resolution and low resolution sequences.

Applying calculus of variations, a minimizing pair $(u, \vec{v})$ must under mild regularity assumptions satisfy the condition $\nabla E(u, \vec{v}) = 0$ where ∇ is the gradient. The optimized solution is expressed by the coupled system of equations

$$\begin{cases} \dfrac{\partial E}{\partial u}(u, \vec{v}) & = 0 \\[2ex] \dfrac{\partial E}{\partial \vec{v}}(u, \vec{v}) & = 0. \end{cases} \tag{4.5}$$

where $\partial E(u,\vec{v})/\partial u = 0$ is subject to the constraint $\mathcal{R}u = u_0$, the projection onto the true solution hyperplane. There is no back projection of the flow, $\mathcal{R}\vec{v} = \vec{v_0}$ as there is no ground truth $\vec{v_0}$ governing what the true solution hyperplane is. When calculating the HR flow $\vec{v_0}$ is linked to the ground truth through its coupling to and dependency of u and the projection $\mathcal{R}u = u_0$.

4.3.3 Variational Video Super Resolution and Optical Flow

It has already been mentioned in the previous sections of this chapter that we have chosen to use total variation as the distribution for all terms. Total variation fits our purpose of balancing between computational complexity and mathematical tractability. Using total variation on the (spatio)temporal parts, E_2 and E_3 in (4.4), allows for temporal edges, which means that occlusions will be handled. It has been discussed in Sections 2.6 and 3.3.2 (Deinterlacing chapter) on which measures of the image sequence intensity values (u) and its flow field ($\vec{v}$) to apply total variation.

Using total variation we get the following energy from (4.4):

$$\begin{cases} E(u,\vec{v}) = & \underbrace{\lambda_1 \int \psi(|\nabla u|^2)dx}_{E_1} + \underbrace{\int \psi(\lambda_2 \left|\pounds_{\vec{V}} u\right|^2 + \gamma \left|\pounds_{\vec{V}} \nabla u\right|^2)dx}_{E_2} + \\ & \underbrace{\lambda_3 \int \big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2)\big)dx}_{E_3} \\ \underbrace{\mathcal{R}u = u_0}_{E_0} & \end{cases} \tag{4.6}$$

where x runs over the whole domain of the HR sequence u, the λ_i's and γ are some positive constant weights and ∇ is the 2D spatial gradient. ∇_3 in E_3 is the 3D local spatiotemporal prior on the flow: v_1 and v_2 are the x- and y-components of the flow field, i.e. $\vec{v} = (v_1, v_2)^T$ and $V = (\vec{v}^t, 1)^T$. $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is an approximation of the $|\cdot|$ function regularizing it around the origin, where it is non-differentiable. ε is a small positive constant set to 10^{-8} in our tests.

E_1 is then the well-known 2D spatial total variation filter and E_3 is a similar 3D regularizer on the flow. E_2, which acts as the spatiotemporal prior on the intensities and likelihood for the flow, is more complex. The two $\pounds_{\vec{V}}$-terms are the $\vec{V}$-directional derivatives of u and ∇u respectively, also known as the Lie-derivatives (see for instance the book on Riemannian geometry by Gallot *et al.* [42]). To set the notation straight we have[4]

$$\pounds_{\vec{V}} u = \frac{\partial u}{\partial \vec{V}} = \nabla u \cdot \vec{v} + u_t = \vec{V}^T \nabla_3 u \approx u(\mathbf{x}, t) - u(\mathbf{x} + \vec{v}, t+1) \tag{4.7}$$

where the right hand side of the approximation is the brightness constancy assumption (constant intensity along the optical flow field) and the other terms are all notions for the linearized version of the brightness constancy assumption know as the optical flow constraint (OFC) [48]. $\nabla_3 u$ is the spatiotemporal gradient of u. The second term in E_2 is

$$\pounds_{\vec{V}} \nabla u = \frac{\partial \nabla u}{\partial \vec{V}} \approx \nabla u(\mathbf{x}, t) - \nabla u(\mathbf{x} + \vec{v}, t+1) \tag{4.8}$$

[4] $\partial f(x)/\partial \vec{V} = \lim_{\epsilon \to 0} \;\; f(x + \epsilon \vec{V}) - f(x) \;\; /\epsilon = \nabla f(x) \cdot \vec{V}$.

where the right hand side of the approximation is the gradient constancy assumption (GCA) and the two other terms are a linearized version of the GCA, which sharpens motion boundaries and lowers sensitivity to changes in brightness (change of lighting, motions in and out of regions in shadow). The notation $\partial \cdot / \partial \vec{V}$ is used to some extent in literature on image and shape geometry but is problematic, as e.g. $\partial \vec{V} / \partial \vec{V} \neq 1$ although it would be considered equal to one in standard (partial) differential notations. Thus to avoid confusion we do not use this notation.

We split (4.6) in two parts, $E(u)$ and $E(\vec{v})$ to minimize it according to (4.5). For the flow we then get this energy to be minimized:

$$E(\vec{v}) = \underbrace{\int \psi(\left|\pounds_{\vec{V}} u\right|^2 + \gamma \left|\pounds_{\vec{V}} \nabla u\right|^2) dx}_{E_2} + \underbrace{\lambda_3 \int \big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2)\big) dx}_{E_3}. \tag{4.9}$$

According to the survey in [12] this energy is the scheme yielding the most precise optical flow. It was first presented by Brox *at al.* in [9]. Simpler versions of variational optical flow algorithms have been shown to run real time on standard PCs in [11], which shows why we are not that far from real time variational VSR even when using variational optical flow methods (the flow calculation is the computationally heavy part of our VSR algorithm).

The intensity energy to be minimized is

$$E(u) = \underbrace{\lambda_s \int \psi(|\nabla u|^2) dx}_{E_1} + \underbrace{\lambda_t \int \psi(\left|\pounds_{\vec{V}} u\right|^2) dx}_{E_2}, \qquad \underbrace{\mathcal{R}u = u_0}_{E_0} \tag{4.10}$$

where we have removed the GCA in E_2 as it gives fourth order terms in the Euler-Lagrange equations of $E(u)$ and makes the computations to minimize the energy very heavy. Since we already have well-segmented flow field with sharp motion boundaries from using the GCA in $E(\vec{v})$, it will most likely not lift the output quality if used in the intensity part as well. Imagine an object moving into a darker region (e.g. a car driving from the sun into the shadow) and take a point p on the object, which is in the sun in frame no. 1, and in the shadow in frames 2 and 3. Temporal diffusion in p in frame 2 along the flow using just the brightness constancy assumption will be from frame 3 mainly as an temporal edge is detected between frame 1 and 2 in p. Adding the GCA will force the gradient to be diffused from both frame 1 and 2, which might lead to an increase in details, but most likely does nothing as we already get precise information from frame 3 alone. The GCA has already done its work in the flow energy minimization. Further discussions on whether or not to use the GCA in $E(u)$ can be found in Chapters 2, 3 and 5 of this thesis.

For details on the Euler-Lagrange equation of (4.9) we refer to Chapter 3 of this thesis and the references [9] and [64]. The Euler-Lagrange equation of (4.10) is

$$\begin{cases} \dfrac{\partial E}{\partial u} = -\lambda_s \mathrm{div}_2 \left(\psi'(|\nabla u|^2) \nabla u\right) - \lambda_t \mathrm{div}_3 \left(\psi'(|\pounds_{\vec{V}} u|^2)(\pounds_{\vec{V}} u)\vec{V}\right) = 0 \\ \\ \mathcal{R}u - u_0 = 0 \end{cases} \tag{4.11}$$

where div_2 and div_3 are the 2D and 3D divergence operators respectively. and the discretization of the optical flow constraint is suggested by the approximation $\pounds_{\vec{V}} u \approx u(x+v, t+1) - u(x,t)$ from (4.7).

4.3.4 Simultaneous Optimization of High Resolution Flows and Intensities

It is the level of integration in minimizing $E(u)$ and $E(\vec{v})$, how the coupled system in (4.5) is solved, that decides the level of simultaneousness of the solution. The high level of simultaneousness in inpainting in [65] and temporal super resolution in Chapter 5 of this thesis comes from using a multiresolution scheme. Both the flow and intensity energies are minimized on each level in the multiresolution pyramid giving highly reliable initializations for the next level when the scale factor from level to level is small. In the deinterlacing in Chapter 3 and in our early video super resolution in [54] simultaneousness cannot be claimed. In both cases in $E(\vec{v})$ is minimized first, using the interlaced video (u_0) in the deinterlacing and the low resolution input u_0 as u in computing $\vec{v}$. This flow is then used in minimizing $E(u)$. This gives fine results for the deinterlacing case. In the VSR case the low resolution flow is all that is computed solving $\partial E(u, \vec{v})/\partial \vec{v} = 0$, the high resolution flow is just a simple initialization from that LR flow. This HR flow does not allow for the full use of temporal information as it is not detailed enough.

It makes no sense to introduce simultaneousness to VSR by using multiresolution with the LR sequence composing the highest resolution. In the simultaneous VSR algorithm presented in this chapter we will therefore iterate between minimizing each of the two parts in (4.5) at the high resolution directly to provide better and better version of the fixed part currently not under optimization.

This will not give us quite the level of simultaneousness as if we solved the system in a multiresolution setting, but to make multiresolution VSR possible we would either have to drop the super resolution constraint derived from the projection back onto the true solution space, $\mathcal{R}u = u_0$, and thus the solution would no longer be correct, or we would get an extremely complex system if we kept the SR constraint.[5] By iterating and all the time using the latest version of the other, temporarily fixed part we do get a well controlled scheme producing optimized flows and intensities. The HR intensities are nicely locked to their LR counterpart, u_0 through the back projection $\mathcal{R}u = u_0$, but we also keep the HR $\vec{v}$ nicely controlled through its dependency of the latest version of u (which is controlled by the projection).

We tried in a first attempt to just recalculate the HR flows on the outputs of our VSR algorithm from [54] and then run the intensity VSR part again, but we got literally no change in the output HR sequence: The HR inputs (intensities and the following flows) were already optimized, but not to the global minimum but to some stable local minimum to the variational approach. Thus we ran the algorithm for too long and on with to poor a flow, and ended up in a (bad) local minima with a smooth output low on details and sharpness. Here we will simultaneously create high quality HR flows and sequences. Thus the iteration between minimizing $E(u)$ and $E(\vec{v})$ will have to be controlled carefully to avoid

[5]As it will be made clear in Section 4.3.5 using the SR constraint with the typical small scale factors in a multiresolution pyramid will create a very complex solution.

local minima.

4.3.5 The Super Resolution Constraint

As mentioned in the previous section, the super resolution constraint does not go well with multiresolution schemes but before we show why that is, we will first describe the SR constraint, which governs the projection back to the solution hyperplane. It can be put as simple as: *The average pixel intensity over an area of interest is kept constant.* We use the uniform distribution to model the point spread function, which makes R, the discretized version of the projection operator $\mathcal{R}$, a moving average filter.

Replacing the uniform distribution with a Gaussian would complicate the SR constraint modelling as we would no longer be working with area overlaps between two set of rectangular pixels in the HR and LR grids respectively (see Figure 4.2(b)) but a more complicated Gaussian weighed area overlap.

The PSF modelling is the first key element in defining the super resolution constraint, the second key element being the magnification factor. Our model allows for arbitrary, independent magnification factors in both frame height and width (integer to integer size of course). Due to the independence between height and width magnification factors we do not require any preservation of aspect ratio in our super resolution constraint, which is fully on purpose as it enables us to change between different pixel aspect ratios in video, e.g. from PAL widescreen 1:1.422 pixels to square 1:1 HD pixels when going from widescreen SD PAL (576×720) to 720p HD (720×1280). This and the classic 2x2 magnification are the two magnifications we have focussed on in our tests, but we can easily adapt our algorithm to do VSR to any other HD format, e.g. the 1080p format (1080×1920). These magnifications keep us reasonably within the limits of SR ([2] and [69]) but we have also tried doing both 4x4 and 8x8 magnifications to see how far we can take our VSR algorithm.

To make the presentation of the SR constraint straightforward, we first describe the discretized back projection filter R in 1D before we take it to 2D.

1D Super Resolution Filter

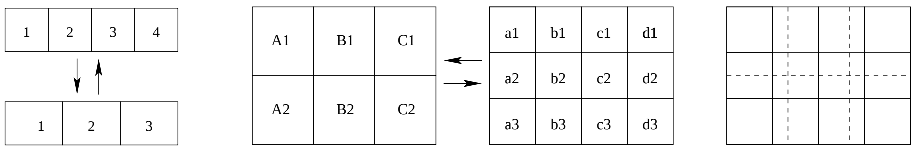
The mapping $R_n^N : \mathbb{R}^n \to \mathbb{R}^N$ can be decomposed as a replication step $S : \mathbb{R}^n \to \mathbb{R}^{nN}$, where each source component is replicated N times, followed by an average step $T : \mathbb{R}^{nN} \to \mathbb{R}^N$, where consecutive blocks of n entries are replaced by their average. A 1D example is shown in Figure 4.2(a). The R_3^4 mapping and its action on a vector $u \in \mathbb{R}^3$ is

$$R_3^4 = \frac{1}{3}\begin{pmatrix} 3 & 1 & 0 & 0 \\ 0 & 2 & 2 & 0 \\ 0 & 0 & 1 & 3 \end{pmatrix}, \quad R_3^4\left(u_1, u_2, u_3\right) = \frac{1}{3}\left(3u_1, u_1 + 2u_2, 2u_2 + u_3, 3u_3\right)$$

which tells us that e.g. the third HR pixel is composed of 2/3 of the middle LR pixel (u_2) and 1/3 of the right LR pixel (u_3).

2D Super Resolution Filter

The 2D and higher dimensional filters of this form are separable, i.e. they can be obtained by cascading 1D filters along the appropriate dimensions. Thus a



(a) 1D (b) 2D

Figure 4.2: The super resolution constraint in 1D and 2D. With uniform PSF it is a matter of calculating area overlap between high resolution and low resolution pixels.

2D filter is obtained by defining the needed vertical and horizontal 1D filters, R_v and R_h, and taking the Kronecker product of them (vectorizing the image matrix lexicographically column by column)

$$R = \frac{mn}{MN} R_h \otimes R_v. \tag{4.12}$$

For the $m \times n = 2 \times 3$ to $M \times N = 3 \times 4$ example shown in Figure 4.2(b) R becomes

$$R = \frac{mn}{MN} R_h \otimes R_v = \frac{1}{6} \begin{pmatrix} 6 & 3 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 6 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 2 & 0 & 4 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 4 & 0 & 2 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 & 6 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 3 & 6 \end{pmatrix}^T. \tag{4.13}$$

This solution tiles for larger integer multiplications of it, so we have to find the greatest common divisors (GCD) of the two sizes (LR and HR) of each of the two dimensions (height and width) respectively. When we e.g. go from 576×720 SD to a 720×1280 HD our projection will map LR pixel blocks of size 4×9 to HR pixel blocks of size 5×16 separated where boundaries of the LR and HR pixel grid aligns. Still, we have 80 HR pixels depending on each others update in this case.[6]

Projecting back to the solution space can be done directly in high resolution: Each time we get an update of the energy, we orthogonally project this energy update back onto the true solution hyperplane T as done for feature based, variational image reconstruction in [68] by

$$\nabla E(u) \perp T = R(R^T R)^{-1} R^T \nabla E(u) \tag{4.14}$$

where u and R are the same as before, $\nabla E(u)$ is the suggested energy update, R^T is the transpose of R and $\nabla E(u) \perp T$ is the correct, back projected energy minimizing update as dictated by the super resolution constraint.

[6]Using Gaussians to model the PSF instead of the uniform distribution would make this divide and conquer block processing strategy impossible as soon as the Gaussian used are larger than 1 pixel wide in its numerical implementation: There would no longer be any boundaries without pixel overlap at which to divide into blocks.

Multiresolution VSR and the super resolution constraint

For the example of going from $m \times n = 2 \times 3$ to $M \times N = 3 \times 4$ in Figure 4.2(b), the projection matrix $M = R(R^T R)^{-1} R^T$ is of size 12×12 and in the SD to HD example it is of size 80×80. Doing 2x2 VSR in a multiresolution setting having a typical scale factor of 1.04 would give some very large and complex mapping matrices at each level, thus multiresolution VSR would be computationally very costly. It is not impossible that multiresolution VSR using small scale factor could improve the output quality. We have done no test using small scale factors, but have done multiresolution VSR with a scale factor of two, which gives a simple 2×2 projection matrix, to get 4x4 and 8x8 VSR.

Combined Spatio-Temporal Video Super Resolution

Discussing multiresolution VSR also brings up the question of combining temporal and spatial video super resolution as it is done in [92] using a multi-camera approach (discussed in Sections 4.2.2, 5.2.1 and 5.2.7 of this thesis). Our framework allows for the combination in theory as well, but in practice it would have to be a cascaded function first doing TSR first and then naturally continuing with VSR when TSR reaches the finest level of its multiresolution solver pyramid, which is the spatial input resolution and the stating point of VSR. The cascading could also be done VSR first followed by TSR. We will discuss integrated solutions in Section 7.2.4.

4.3.6 Minimizing Algorithm

Initialization

Choosing the right initialization for our algorithm is important. In Section 4.3.4 we described how trying to rerun our VSR algorithm from [54] failed as the rather smooth inputs where considered optimized and left us stuck in a local minimum. Bilinear interpolation produces very smooth results and using it as initialization would also leave us caught in a local minima. Thus we use the projection in (4.12)

$$R = \frac{mn}{MN} R_h \otimes R_v$$

to initialize both flows and intensities to get unblurred (but also jagged) initializations. We have tried (with our non-simultaneous VSR algorithm from [54]) to initialize the intensities with bilinear interpolation instead, and as predicted we got stuck in a local minimum with outputs so smooth that they were hardly different from their inputs. The fact that our variational method is satisfied with blurred regions is why motion blur will also stay untouched doing VSR. Thus we do not change the artistic work of the film makers. Algorithms using structure tensor regularization also cannot de-blur already (too) blurred regions as it is pointed out by Roussos and Maragos in [86].

If we start with bilinear interpolation of the HR flows using the non-simultaneous VSR algorithm from [54] it does not change the intensity output, which is a clear indication that an accurate and reliable HR flow is necessary to get optimal VSR results. The LR flow that we initialize with R is computed on the input LR sequence using a full multiresolution scheme with the same type of fixed point approach as described below for the intensities. It is the energy in

(4.9) that is minimized, but with the modification that u is replaced by the low resolution sequence u_0 such that x only runs over the LR domain. To calculate the backward LR and HR flows on the sequence, the energy in (4.9) is minimized on a time reversed ($\{1, 2, 3..., n\} \mapsto \{n, n-1, n-2, ..., 1\}$) version of u_0 and u respectively.[7]

Core Algorithm

To efficiently minimize the energy in (4.10) we handle the nonlinearity of its Euler-Lagrange equation (4.11) by using a fixed point approach. First we define $A(u) = 2\psi'(|\nabla u|^2)$ and $B(u) = 2\psi'(|\pounds_{\vec{V}} u|^2)$, which represents the nonlinear parts of (4.11). We only update $A(u)$ and $B(u)$ in a number of outer fixed point iterations in order to have a fully linear approximation of (4.11). For each outer fixed point iteration, we run a number of inner iterations on the now linearized system using a Gauss-Seidel relaxation scheme with the super resolution constraint incorporated: We back-project the energy update in each inner iteration modulated by a positive weight, α, in order to respect intensity bounds. A similar solver is used to minimize the energy of the flows in (4.9). The overall sketch of the algorithm is:

- Let u^0 be the initialization of the HR sequence, and
- let $\vec{v^0}$ be the initialization of the HR flow.
- For $i = 0$ until $I - 1$
  1. Run L outer fixed point iterations each with M inner relaxation iterations of the flow solver.
  2. Run J outer fixed point iterations each with K inner relaxation iterations of the intensity solver.
  3. In each solver always use the latest updated version of the other component (u for $\vec{v}$-solver and the other way around).
- Output the final HR sequence u, and if needed, the final flow $\vec{v}$.

The intensity solver used above is given next. Denoting the null space of R by T as in Sec. 4.3.5 and by P_T the orthogonal projection onto T as in (4.14), we have:

- Let u^0 be the initial guess for the HR sequence.
- For $j = 0$ until $J - 1$
  1. Let $A^j := A(u^j)$, $B^j := B(u^j)$, $u^{j,0} := u^j$.
  2. for $k = 0$ until $K - 1$
     (a) From $u^{j,k}$, compute $\bar{u}^{j,k+1}$ by one Gauss Seidel sweep on $G(A^j, B^j)$.
     (b) Form the update vector $\bar{\delta}^{j,k} := u^{j,k} - \bar{u}^{j,k+1}$.
     (c) Project it onto T, $\delta^{j,k} := P_T(\bar{\delta}^{j,k})$.

[7]The multiresolution scheme in computing the flow employs a down- and upsampling scheme described in [11] and is the same as using R to scale from level to level with rounding to the nearest integer frame size as typical scale factors give non-integer frame sizes.

(d) Update current estimate $u^{j,k+1} := u^{j,k} - \alpha\delta^{j,k}$.

3. $u^{j+1} := u^{j,K}$.

- Output the HR sequence u^J.

Leaving out the projection, the flow solver runs similarly.

Projection Details

As described in Sec. 4.3.5 the projection allows processing in blocks, e.g. in blocks of $5 \times 16 = 80$ pixels in the SD to 720p HD case mentioned, so for any $M \times N$ block step 2(c-d) above is

$$u_l^{j,k+1} = u_l^{j,k} - \alpha \left(\bar{\delta}_l^{j,k} + \sum_{i=1}^{MN} P^T(l,i)\bar{\delta}_i^{j,k} \right), \qquad l = 1, ..., MN \tag{4.15}$$

due to the dependance of the update of all other pixels in the support area. A simple example: In the 2x2 SR case, if the suggested update for pixel 1, $\bar{\delta}_1^{j,k} = 4$ and $\bar{\delta}_l^{j,k} = 0$ for the remaining pixels in the current block ($l = 2, 3, 4$), equation (4.15) tells us to add one to all four pixels and subtract four from pixel 1. This is the normalization incorporated in P^T as dictated by the super resolution constraint.

The variational (diffusion) part of our algorithm follows the minimum-maximum principle (see for instance [80]) and keeps intensity values within bounds, but we need the weight α in (4.15) to ensure that the projection does not create values out of bounds as it does not follow the minimum-maximum principle. If a value in a block, B, is detected out of bounds, α is recursively lowered from 1 to 0 in steps of 0.1, and all values in the block B recalculated with the new α, stopping when all values in B are within bounds. This can potentially stop the evolution of the sequence, but it was found not to do so in practice. The problem of out of bound intensities was limited to very few regions around extremely high contrast edges and thus the computational overhead by enforcing bounds was also negligible.

4.3.7 Discretizations

So far we described the overall algorithm and the solvers used. The final component missing in the complete description of our algorithm is the numerical details in discretizing the Euler-Lagrange equation (4.11). For details on the discretization of the flow Euler-Lagrange of (4.9) we refer to [9] and [65].

We first look at the spatial term of (3.15) and following the ideas of discretization of total variation on the spatial 2D gradient given in [1], [18], [64] and [88] we can do a simple rewrite

$$\operatorname{div}_2\left(\psi'(|\nabla u|^2)\nabla u\right) = \operatorname{div}_2\left(\frac{\nabla u}{\psi(|\nabla u|^2)}\right) = \operatorname{div}_{xy}(A\nabla u) \tag{4.16}$$

where

$$A = \frac{1}{\psi(|\nabla u|^2)}.$$

The spatial gradient is

$$\nabla u = (\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y})^T = (\partial_x u, \partial_y u)^T. \tag{4.17}$$

Now from the definition of the 2D divergence we have

$$\text{div}_{xy}(A|\nabla u|) = \partial_x(A\partial_x u) + \partial_y(A\partial_y u) \tag{4.18}$$

Using (4.17) and (4.18), equation (4.16) can be approximated finitely as

$$\text{div}_{xy}(A\nabla u) \approx \delta^o_{x,h/2}(A\delta^o_{x,h/2}u) + \delta^o_{y,h/2}(A\delta^o_{y,h/2}u) \tag{4.19}$$

where $\delta^o_{x,h/2}$ is the central difference approximation of $\partial u/\partial x$ and $\delta^o_{y,h/2}$ the same in the y-direction. h is the distance between two horizontally or vertically neighboring pixel positions (grid points) and in most cases set to 1. This is in coherence with the fact, that these filters are only applied to 1:1 square HD pixels. By definition

$$\begin{aligned}\delta^o_{x,h/2} =& \frac{(u(x+h/2,y,t) - u(x-h/2,y,t)}{h} \\ \delta^o_{y,h/2} =& \frac{(u(x,y+h/2,t) - u(x,y-h/2,t)}{h}.\end{aligned} \tag{4.20}$$

Before we do the next rewrite we switch to compass coordinates where the indexes w, e, n, s represents the directions in the grid and is short for west, east, north and south, $u_w = u(x-1, y, t)$ and so on. u_c is the center (or current) pixel. Plugging (4.20) in (4.19) and rewriting gives us the divergence approximation

$$\begin{aligned}\text{div}_{xy}(A\nabla u) \quad \approx \quad & \frac{1}{h^2}\Big((A_{w/2} \cdot u_w + A_{e/2} \cdot u_e + A_{n/2} \cdot u_n + A_{s/2} \cdot u_s) - \\ & (A_{w/2} + A_{e/2} + A_{n/2} + A_{s/2}) \cdot u_c\Big).\end{aligned} \tag{4.21}$$

We need the A's at half-grid point, and we get them by calculating A's at all grid points and then interpolating them at the half-grid points. We could use linear interpolation, i.e.

$$A_{w/2} \approx (A_c + A_w)/2 \tag{4.22}$$

but

$$A_w \approx \frac{2}{\frac{1}{A_c} + \frac{1}{A'_w}} \tag{4.23}$$

endorses edge preservation. Say $A_c = 100$ (flat region) and $A'_w \approx 0$ (edge), then the approximation would be $A_w = 50$ from (4.22) corresponding to a flat region and $A_w \approx 0$ from (3.25) corresponding to the close by edge we wish to preserve. To calculate the A's at grid points, the choice of $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is clever. Since $1/|\nabla u| = 1/\sqrt{(\partial_x u)^2 + (\partial_y u)^2}$ the choice leads to

$$A \approx \frac{1}{\sqrt{\frac{1}{4}\big((u_e - u_w)^2 + (u_n - u_s)^2\big) + \varepsilon^2}}$$

when using central differences and is similar to the 2D total variation term used for inpainting by Chan and Shen in [18]. Extensive testing of this 4-point discretization and two more advanced 8-point discretizations in [57] showed no significant difference in output quality when used for deinterlacing, and thus we use this 4-point version as it approximately a factor of two faster than the two 8-point versions.

For the temporal part, the approximation $\pounds_{\vec{V}} u \approx u(x+v, t+1) - u(x,t)$ gives the discretization and we end up with a 1D version of (4.16) but with the derivative given along the flow. Since the flow is subpixel accurate, the intensities at the position the flow points to is mostly not at grid point and has to be interpolated. We use bilinear interpolation for that.[8] This might be considered an Achilles heel of the scheme, but the fact is that any information on details have to found in or very close to grid points to exist at all makes the bilinear interpolation as reasonable choice. We will use bilinear interpolation until we can easily a) do VSR on flexible grids b) come up with other subpixel interpolation computationally cheaper and more accurate than bilinear interpolation, or c) use some other method, e.g. let the flow continue for several frames until it gets close to grid point. u_c is once again the current, center pixel, $u_a = u(x+v_x, y+v_y, t+1)$ is the after intensity value along the forward flow and $u_b = u(x+v_x, y+v_y, t-1)$ is the before intensity value along the backward flow and we have

$$B_a \approx \frac{1}{\sqrt{(u_a - u_c)^2 + \varepsilon^2}} \quad \text{and} \quad B_b \approx \frac{1}{\sqrt{(u_c - u_b)^2 + \varepsilon^2}}$$

This lead us to the complete direct solution of the scheme – the Gauss-Seidel solver

$$u_c = \frac{\lambda_1(A_w \cdot u_w + A_e \cdot u_e + A_n \cdot u_n + A_s \cdot u_s) + \lambda_2(B_a \cdot u_a + B_b \cdot u_b)}{\lambda_1(A_e + A_w + A_n + A_s) + \lambda_2(B_a + B_b)}. \quad (4.24)$$

A Boundary Conditions Issue

The boundary conditions (BC) of our VSR algorithm is inherited from the inpainting algorithms presented in [64], where a theoretical discussion on the subject of boundary conditions can be found. We use Dirichlet BC on the flow when used in the intensity calculations that is, if there is no flow value available (primarily forward flow in the last frame of a sequence and backward flow in the first frame of a sequence), the flow is set to zero. This is logically the only thing to do as there is no information to find along these flows and we will thus get a zero contribution from this component. For the intensity values we – as it is often the case in image processing – use Neumann boundary conditions to prevent diffusion across the sequence boundaries. This use of boundary conditions, which is perfectly logical and correct for still images, is not nearly as well suited for image sequences. The temporal nature of image sequences is not handled well by Neumann BC as there is actually a flow of intensities in and out of the spatial camera framing of the scene.

Here we will analyze one particular problem occurring in the first and last frame of any sequence, and another problem of using Neumann BC at the spacial boundaries of the image sequence will be discussed in Section 5.4.

[8] Using bicubic interpolation instead did not improve results.

Looking at (4.24) and leaving out the spatial part for now, we have

$$u_c = \frac{B_f \cdot u_a + B_b \cdot u_b}{B_f + B_b}.$$

Say we are in the last frame (the process is similar for the first frame) and we use Neumann boundary conditions for the intensities in combination with Dirichlet BCs for the flows, then $u_a = u_c$ (flow is zero and u_c is on the forward temporal boundary of the data volume). This leads to

$$B_f = \frac{1}{\sqrt{(u_a - u_c)^2 + \varepsilon^2}} = \frac{1}{\sqrt{(u_c - u_c)^2 + \varepsilon^2}} = \frac{1}{\sqrt{\varepsilon^2}} = \frac{1}{\varepsilon} \tag{4.25}$$

and

$$u_c = \frac{B_f \cdot u_c + B_b \cdot u_b}{B_f + B_b} \tag{4.26}$$

Combining (4.25) and (4.26) gives us

$$u_c = \frac{\frac{1}{\varepsilon} \cdot u_c + B_b \cdot u_b}{\frac{1}{\varepsilon} + B_b}$$

$$\Updownarrow$$

$$u_c(\tfrac{1}{\varepsilon} + B_b) = \tfrac{1}{\varepsilon} u_c + B_b u_b$$

$$\Updownarrow$$

$$u_c B_b = B_b u_b$$

$$\Updownarrow$$

$$u_c = u_b \tag{4.27}$$

This is of course a very strong update of the pixels in the last frame (and the first frame where we get $u_c = u_a$). The spatial part was left out but with the result obtained in (4.27) the spatial part will have to produce a very strong contribution to the update to get any influence. Not even when using a spatial to temporal diffusion ratio of $\lambda_s : \lambda_t = 500 : 1$ did the spatial part manage to dominate over the temporal part and the first and last frame of the sequence in test would stay practically unchanged from their initialization.

One might consider this a major flaw in the implementation, but it is in line with standard uses of boundary conditions and the problem is easily fixed in practice. One can either include an extra frame in each end of the sequence being processed or simply set $\lambda_t = 0$ in the first and last frame of the sequence, thus only using spatial diffusion in these frames. To keep the computational cost down we used the second approach. The problem will not have an effect at cuts in the sequence, but just provide a very strong temporal (occlusion) edge. This claim was found to be true empirically when running a test on a substantial amount of sequences with cuts.

4.4 Experiments

Before we evaluate the results obtained with our video super resolution algorithm there are some topics we would like to go through.

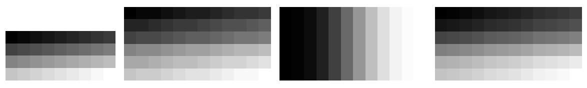


Figure 4.3: The importance of the super resolution constraint. (a) input, (b) HR initialization, (c) VSR without SR constraint destroys the image content, and (d) preservation of true content when running VSR with the SR constraint. The HR initialization (b) and the result with SR constraint (d) are very similar as there is no sharp edges in this example.

4.4.1 The Importance of the Super Resolution Constraint

We have stressed the importance of using the super resolution constraint throughout this chapter and would like to give an example why we add this extra complexity to our already complex regularization scheme. In Figure 4.3(a) we have shown one frame of four identical LR frames in a small sequence, a 4×9 matrix filled line by line with the values 1 to 36, and in Figure 4.3(b) the corresponding HR frame, a 6×12 matrix initialized using (4.12). In Figure 4.3(c) we see how pure regularization without the SR constraint destroys the image content, whereas it is preserved in Figure 4.3(d) with the back projection of the SR constraint included. The breakdown shown in Figure 4.3(c) is similar to the classical cartoon effect obtained when running 2D spatial total variation for too long on a natural image. When we increase the weight for the temporal diffusion in our VSR algorithm (knowing the flow to be zero) the destruction process is significantly slowed down. In practice we are however unable to discard the SR constraint as flows are not (yet) guaranteed to be precise enough for that, and in test we ended up with cartoons when running VSR on real sequences without using the SR constraint.

The advantage of leaving out the SR constraint – besides lowering the complexity and the computational cost – would be de-noising and reduction in blockiness but with an inevitable loss of details. One could chose to update each pixel with a weighted mix of the pure total variation regularization and the SR constraint controlled update. Since we work on clean data and have had no real problems with blockiness, we have not experimented with weighed mixing.

4.4.2 Subjective and Objective Evaluation

We have used both subjective and objective evaluation of our results, but focussed on the subjective evaluation to best mimic the perceptive evaluation in the human visual system. We have discussed objective vs. subjective evaluation in Section 2.3 of this thesis and further information can be found in [4], [51] and [77]. Our view is that objective measures can give an idea about the subjective quality, and in some limited cases (small data set and/or a specific problem) they might even be in almost full correlation with the subjective results. In Chapter 3 we have shown experimentally for the case of deinterlacing how the very popular means square error (MSE), which is also directly equivalent with the peak signal to noise ratio (PSNR), does not correlate with subjective results. Our subjective evaluation have been done by a ourselves and in some case by other image (sequence) processing experts.

The above discussion covers the evaluating of the output video sequences. Evaluation of optical flows quality in literature is typically done by testing the algorithm on artificial sequences with known ground truth flows and measuring the angular error between ground truth and computed flows (see for instance [12]). On real video sequences one is left to visually evaluate the quality of the flows, which we have done extensively to find the optimal parameter settings. Fortunately we produce output intensity sequences based on the flows and as we consider the flows the means for producing intensity sequences, we can further tune our flow calculations by evaluating the quality of the intensity outputs.

4.4.3 Test Material

We aim at applying our VSR algorithm either at the end user in a home video/entertainment system or in the video scaling system at a broadcaster. Therefore we have chosen to conduct our tests using standard video material, specifically PAL DVDs telecined from film. We use a number of 5-15 frames sequences selected to be challenging in terms of detail level and motion complexity. We work only the luminance channel (8 bit, [0-255]) of the test sequences, but since the two chroma channels are already subsampled (the HVS is less sensitive to details in the color channels than in the luminance channel [70]) in practically any broadcasting or storage system today, simple bilinear interpolation can be used here – at least at lower magnification factors. Of course our algorithm can be applied to the C_r and C_b color channels as well or on a RGB version of the sequence. Some ideas on how to couple the processing of the three channels can be found in [103].

4.4.4 Parameters

As with almost any other image/video processing algorithm we have a number of parameters that needs to be tuned. We have run extensive tuning tests, but with nine free parameters it is of course not complete. Testing just three different settings of each parameter in all possible combinations would result in $3^9 = 19683$ different test results for evaluation. Thus we rely on our common sense and our experience with variational methods for inpainting ([64] and [65]), deinterlacing (Chapter 3 of this thesis) and prior work on variational VSR [54] and have (hopefully) optimized the parameters for use on video data in general.[9] Unless clearly stated otherwise, the parameter settings given in this section are the ones used in all tests.

The initial low resolution flow is calculated running 10 fixed point iterations each with 40 inner relaxation iterations. 5 times 20 iterations or even less will give the same results visually, but to be on the safe side we ran 10 times 40 iterations in our tests. The multiresolution pyramid has 100 levels with a coarse-to-fine scale factor of 1.04. The weight on the gradient constancy assumption in (4.9) is $\gamma = 200$ and the weight on the prior is $\lambda_3 = 70$.

The actual VSR algorithm runs 10 outer iterations. For both the flow and intensity calculations respectively we run 1 fixed point iteration with 5 inner

[9]We do not believe there is any data available in any major broadcast or storage format recorded or artificially generated, on which or settings would fail. There might be cases where changing the settings would produce slightly better results.

relaxation iterations. For the flow $\gamma = \lambda_3 = 100$ in (4.9) and for the intensity calculations $\lambda_s = \lambda_t = 1$ in (4.11). In all computations the convergence threshold is set to 10^{-7} (and never reached).

On our way to the optimal parameters for the actual VSR algorithm given here, we made a few interesting discoveries.

Increasing the number of outer overall iterations does not give any improvements, while lowering the number from 10 to anywhere down to five can give just as good results, but 10 is the failsafe setting. The algorithm is fairly sensitive to changes in the number of inner iterations (fixed point and relaxation). Iterating to much on either the flow or the intensities stops the other from evolving further: Probably a local minimum is reached. Lowering the number of inner iterations causes a slowdown in convergence (the system is not sufficiently relaxed). It seems there is a fragile balance between evolving the flow and the intensities, especially the flow should not be allowed to evolve too far without the intensities being evolved simultaneously as the flow then seems to get stuck in local minima resulting in a loss of details.

Changing the γ and λ_3 weights of the flow mainly changes how homogeneous the flow is, but larger changes from the optimal settings results in either too smooth or too detailed intensity outputs, too detailed meaning artifact-like details or oversharp edges appearing clearly unnatural to the sequence ("they should not be there").

Increasing the spatial diffusion by turning up λ_s gives smoother results similar to the ones obtained with the nonsimultaneous VSR algorithm from [54] (comparisons given in Section 4.4.7). Turning up λ_t has no effect on some sequences and on others it slowed down development away from the initialization. It seems the implicit weighing in the variational algorithm is enough to ensure optimal temporal diffusion and pushing it too hard with high λ_t-values is unnecessary or even has a negative effect. We also experimented with changing λ_s and λ_t over time (e.g. eight outer iterations with $\lambda_s = \lambda_t = 1$ and two with $\lambda_t = 5$) and got minor improvements on some sequences, whereas the same settings failed on other sequences. Thus finer parameter tuning might improve some results, but it is data dependent and we have no measure to automatically control it. Since we aim at general applicability of our algorithm, we use the fixed set of failsafe parameters as given above.

4.4.5 Running Times

We have not focussed on optimizing our code for speed (yet). The code is written in C++ using the CImg library (`http://cimg.sourceforge.net/`) interfacing with Matlab.

The initial flow computations (backward and forward) takes 1-4 hours on a standard PC. The running time of the VSR algorithm when doing 2x2 magnification from 576×720 SD PAL resolution is ca. 19 seconds per outer iteration per frame. from 576p SD PAL to 720p HD (720×1280) the running time is ca. 13 seconds per outer iteration per frame on the same PC. The number of HR pixels being processed is 1.7 times higher in the 2x2 case but the processing time is only 1.45 times higher, which shows that the more complex back projection in the SD to HD case (80 corrections per pixels contra 4 in the 2x2 case) does give a minor overhead in computation time.[10] The times given are on a specific sequence but there is no major difference in average computation time per pixel from sequence to sequence: The algorithm is in practice data independent when it comes to running times. It is clear that it is in the initial flow computations the need for speedup is the greatest, but from [11] we know that simpler variational flow algorithms run in real-time on standard PCs (although on web camera resolution video).

4.4.6 Online Material and Correct Viewing of Results

Selected test results are available as video (*.avi) and electronic stills (*.bmp) online at: `http://image.diku.dk/sunebio/VSR/VSR.zip`. The printing process will often blur the figures, so the results given as figures in this chapter are best viewed on-screen and in some cases with a given zoom (given in the captions). In Appendix A the figures requiring zooms are given at the zoomed size, but we still recommend on-screen viewing. This is to best see the correct results and to some degree to simulate a true 1:1 resolution relation between image and screen. The program `Virtual Dub` that displays avi (and bmp) files at their true 1:1 resolution is included in the zip file.

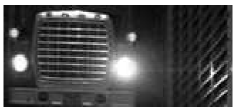
4.4.7 2x2 and SD to 720p Results

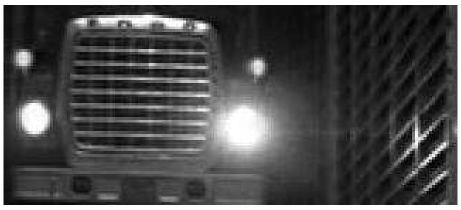
In this section we will focus on subjective evaluation of results from doing VSR with the classic 2x2 magnification and 576p SD PAL to 720p HD VSR (576×720 to 720×1280). In Section 4.4.8 we will give objective results and in Section 4.4.9 we will give 4x4 and 8x8 VSR results. There exist no commonly used benchmark for evaluation of (video) super resolution results and the only thing most tests have in common is the use of the magnification factor 2. The general problems of benchmarking is discussed in Section 5.2.8 of this thesis.

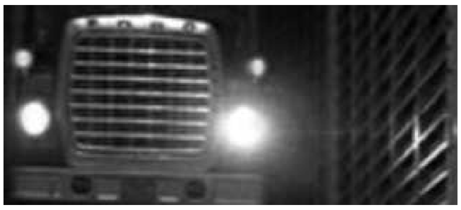
SR algorithms with learned priors are only applicable on specified types of data (e.g. faces or text as in [2]) they cannot be used in processing of general video. On the data they are trained for they are likely outperform our VSR algorithm and structure tensor based SR methods would also be very competitive, but they are for now to computationally heavy for our field of application. Methods like to one given by Farsiu *et al.* in [38] are typically limited to simple flow (registration) modelling, thus making them unfit for use on general video signals. Thus we have chosen to compare our results to our own nonsimultaneous VSR results, to bilinear interpolation, the most advanced(!) method widely used in video processors today, and to bicubic interpolation, which is know to perform better than bilinear interpolation.

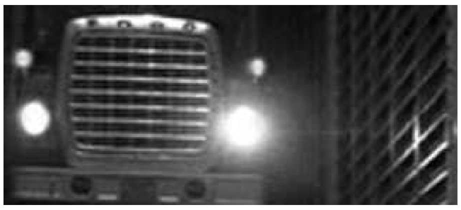
Results for 2x2 SR/VSR on the sequence `Truck` is given in Figure 4.4 (zoomed versions in Figures A.1 and A.2 in Appendix A). First we see how jagged (or blocky) the Initialization is in 4.4(b) and how bilinear interpolation produces a very smooth result in 4.4(c). Bicubic interpolation produces a much sharper result as seen in 4.4(d) but the output of our two variational VSR methods in 4.4(e) and 4.4(f) are even sharper again. The simultaneous VSR (S-VSR) produces a sharper result than the nonsimultaneous VSR but the difference between the two are not as big as the differences bilinear to bicubic or

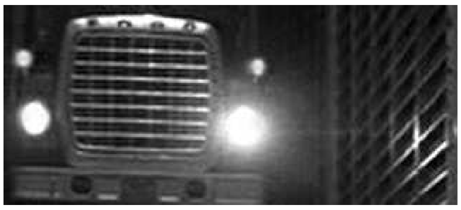
[10]With reservation for different handling of the jobs by the computer (memory allocation etc.).


(a) LR input


(b) HR 2x2 initialization


(c) Bilinear 2x2 SR


(d) Bicubic 2x2 SR


(e) Nonsimultaneous 2x2 VSR from [54]

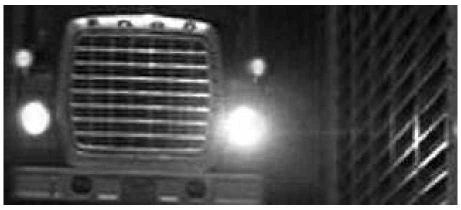
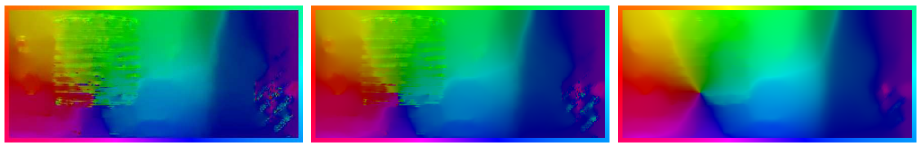

(f) Simultaneous 2x2 VSR (S-VSR)

Figure 4.4: 2x2 VSR on the sequence `Truck`. A 70×160 pixels cutout of LR is shown in (a) and the HR cutouts (b)–(f) are 141×321 pixels. Sharpness increases step by step from (c) through to (f) as seen most clearly when viewed on-screen zooming to 175% or more. (a)-(f) can be found zoomed in Figures A.1 and A.2 in Appendix A. For `Truck` the motion is a zoom as the truck drives towards the camera.

bicubic to nonsimultaneous VSR. The videos from which Figures 4.4(c), 4.4(e) and 4.4(f) are taken from are given in the online material [59] in the folder `Truck_2x2_oldAndNewVSRandBilinear`.

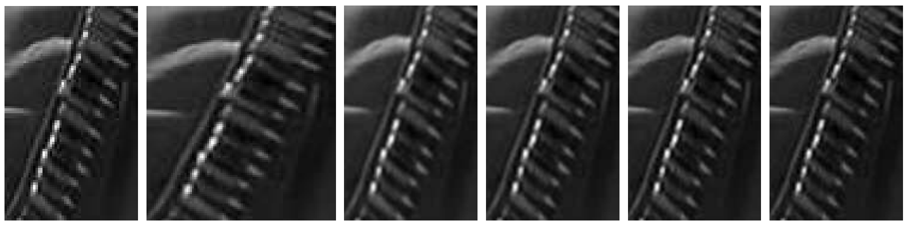
The flow produced using the optimal settings given in Section 4.4.4 is shown in Figure 4.5(b). In Figure 4.5(a) we see how lowering the weight on the prior of the flow from $\lambda_3 = 100$ to $\lambda_3 = 70$ makes the flow a bit oversegmented, resulting in artifacts in the intensity result (bright single spikes and horizontal lines on the front grill). The very smooth flow resulting from setting $\lambda_3 = 250$ and shown in Figure 4.5(c) seems more correct (i.e. the grill line flows are now part of the overall zoom and not segmented independently), but the intensity output loses in sharpness. This illustrates the approach we used in parameter tuning: We always tried to find the balance point between introducing artifacts and over-smoothing.

As with `Truck` we also see a gradual improvement in sharpness from bilinear over bicubic and nonsimultaneous VSR to simultaneous VSR on all other 18 sequences in our test when doing 2x2 magnification. For some sequences like `Bullets` shown in Figure 4.6 the differences are less significant. The sequence `Truck` appears very sharp in its LR version while `Bullets` appears less sharp



(a) Oversegmented flow (b) Flow with optimal settings (c) Oversmoothed flow

Figure 4.5: Flows from 2x2 VSR on the sequence `Truck`of a truck driving towards the camera. The direction of the flows is given by the hue value (in color) on the border and the magnitude by the intensity.



(a) LR (b) 720p S-VSR (c) Bilinear (d) Bicubic (e) VSR [54] (f) S-VSR

Figure 4.6: VSR on the sequence `Bullets`. Cutout sizes are: 90×56 (LR), 113×100 (720p) and 179×111 (2x2). To illustrate the increase in detail level with higher resolution the LR input (a) and the 720p VSR result (b) are shown at the same height as the 2x2 results in (c) to (f). As with `Truck` we have increasing sharpness step by step from (c) to (f) although less significant than for `Truck`. Results are best viewed on-screen zoomed to 200% or more as in Figure A.3 of Appendix A. The 720p result is shown at its correct aspect ratio whereas the rest (LR and 2x2) are 1:1.422 PAL anamorphic widescreen pixel shown here as 1:1 pixels.

(due to recording method or choices in post processing and the film to DVD transfer processing etc.) and overall the differences between the results of the algorithms are more significant in the test sequences containing details to begin with: There is more information available for temporal (and spatial) transport and less trouble from ending up in local minima due to already smooth regions of image data. When doing SD to HD 720p VSR the differences between the algorithms are smaller, the exception being that bilinear always perform much worse than the three other algorithms. Comparing the 720p result on `Bullets` in Figure 4.6(b) with the 2x2 result in Figure 4.6(f) shows how the higher pixel density – e.g. two screens of the same size and viewing distance but with different resolution – carries more information and gives room for larger improvements.

Figure 4.7(b) shows how simultaneous VSR removes the blockiness seen in the LR input in Figure 4.7(a) and Figure 4.7(c) shows how `Boardwalk` would look on many LCD and plasma tv-sets of today when bilinear interpolation removes the blockiness but at the price of smoothing. The Figures 4.7(d) and 4.7(e) shows just how big the gain in detail from SD to 720p HD can actually be. In the zooms on `Manhattan` shown in Figures 4.8(d) and 4.8(e) there is no difference between simultaneous VSR and bicubic interpolation on the heli-

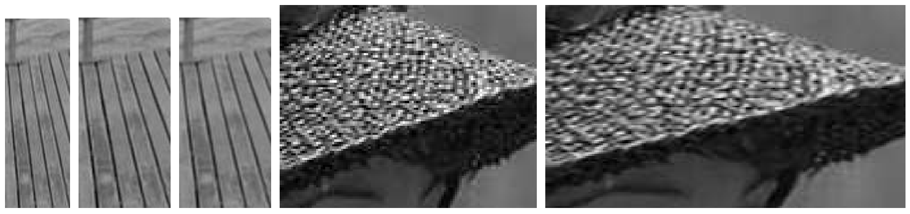


Figure 4.7: 720p VSR on the sequences `Boardwalk` and `Straw Hat`. The blockiness in the cracks between the boards in `Boardwalk` is removed going from (a) LR SD to HR 720p HD, in (b) by simultaneous VSR and in (c) by bilinear interpolation, where (b) is still sharp. On `Straw Hat` a major gain in details is obtained going from (d) SD to (e) 720p HD. Best viewed on-screen zoomed to 150% as in Figure A.4 of Appendix A. Cutout sizes are `Boardwalk`: 146×50 (LR) and 182×88 (720p) and `Straw Hat`: 113×89 (LR) and 113×201 (720p).

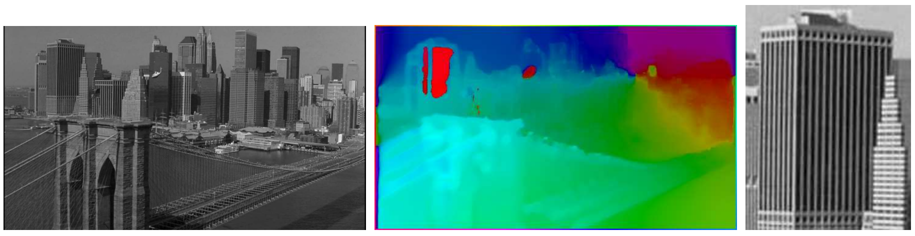
copter, but on the building to the right simultaneous VSR performs a lot better than bicubic interpolation.

The figures given in this chapter do not give the full picture of the differences between the different algorithms as the outputs should be seen as video on a large screens. Local gains in sharpness can to a large extent be evaluated on still, but the sense of overall gain in sharpness of full frames (720×1280 or 1152×1440) is hard to portray in printed figures. To really determine how big an advantage the gain in sharpness is a test with longer sequences, should be conducted under realistic viewing conditions and on large screens. Preferably according to the subjective quality evaluation standard ITU-R Rec. 500 [50] or similar.

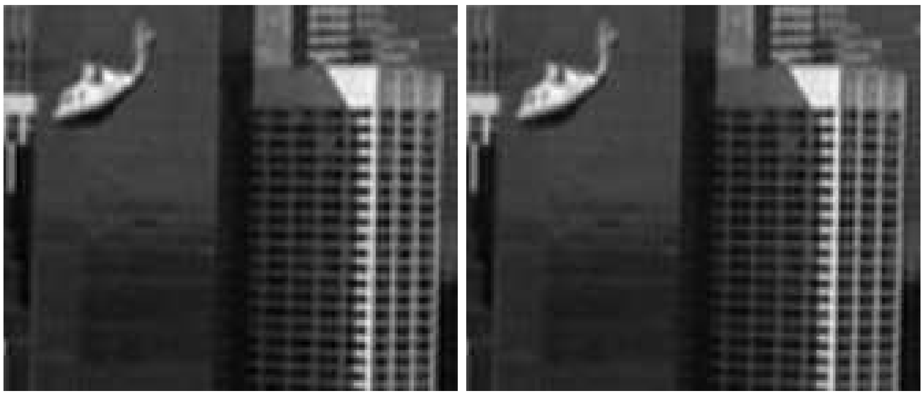
Another improvement only seen when viewing the outputs as video is the decrease in flicker. It is (almost) impossible to compare differences in flicker between LR and HR version of a video as they cover different areas of a given screen and flicker perception is highly dependent on the size of the image projected onto the eye (see Section 2.2 of this thesis). But across different HR results, we can do a comparison. A video example is found in the folder `Boardwalk2_720p_FlickerReduction` of the online material [59], where bilinear, bicubic and simultaneous VSR results are given in 720p HD. The results produced using bicubic interpolation flickers the most, while our nonsimultaneous VSR does significantly better and is close to having as little flicker as the two best algorithms here, simultaneous VSR and bilinear interpolation.

On large and bright screens (e.g. 42 inch plasma and LCD displays) the reduction in flicker is a major quality improvement. As bilinear interpolation smooths outs to many details and edges and the temporal regularization in simultaneous VSR removes flicker while still preserving details and sharpness including the film granularity, there is no doubt that simultaneous VSR produces the best results in SD to 720p conversion and is also best at 2x2 VSR, where the quality difference in the results from the algorithms is more significant.[11] To

[11] As with motion blur, film granularity is a considered an artistic tool, which the film makers wish to preserve.



(a) 720p simultaneous VSR (b) Corresponding flow (c) Flow error handling

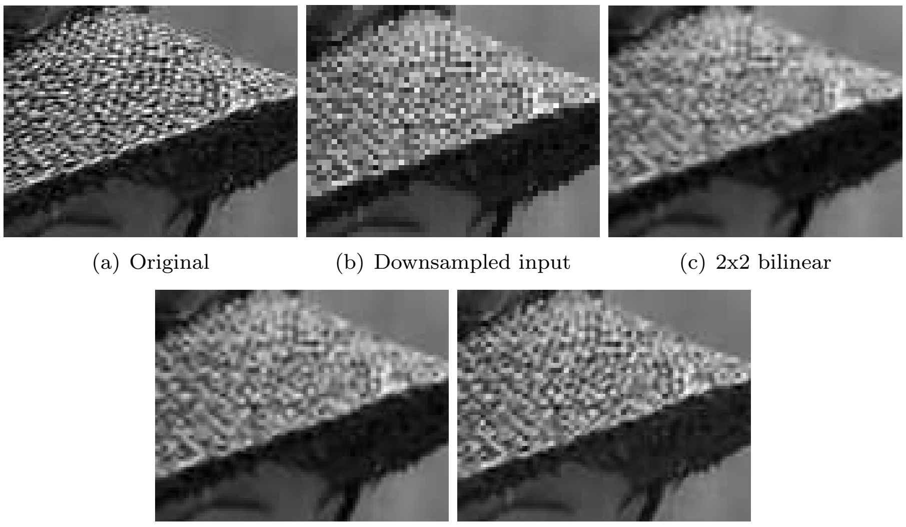


(d) 720p bicubic (e) 720p S-VSR

Figure 4.8: 720p VSR on the sequence `Manhattan Flyby`. A bad case of faulty flow: The repetitive structure of the building on the left in (a) causes the flow calculations to fail as seen in the flow field (b), but in spite of that the result is still good as seen in the zoom-in in (c). (d) and (e) shows the difference between simultaneous VSR and bicubic interpolation in SD to 720p HD conversion. All subfigures are best viewed on-screen and (a)-(c) and are zoomed in Figures A.5 and A.5 of Appendix A. (a) and (b) are full frame 720×1280, (c) is 226×166 and (d)-(e) are 141×166.

stress this point we did 2x2 simultaneous VSR on a 25 frame sequence provided to us by the film post production company Digital Film Lab, and their evaluation of the result was, that is was a lot better than anything they would be able to produce with any of their professional post production and editing systems (Da Vinci, Discreet/Autodesk and others).

Even though variational flow methods produce the most accurate flows available (see the survey in [11] by Bruhn *et al.*) they still fail on occasion. The sequence `Manhattan` in Figure 4.8 is an example of this. The repetitive structures on the buildings cause highly incorrect flows to be computed in several small regions, worst on the building on the left in Figure 4.8(a). Still the intensity output is fine as seen in Figure 4.8(c) due to the robustness of our VSR method: a) if the flow is wrong, the detected structure might be correct, and b) the spatial regularization takes over if the gradient of the intensity along the flow trajectory is too large.


(a) Original (b) Downsampled input (c) 2x2 bilinear

(d) 2x2 bicubic (e) 2x2 simultaneous VSR

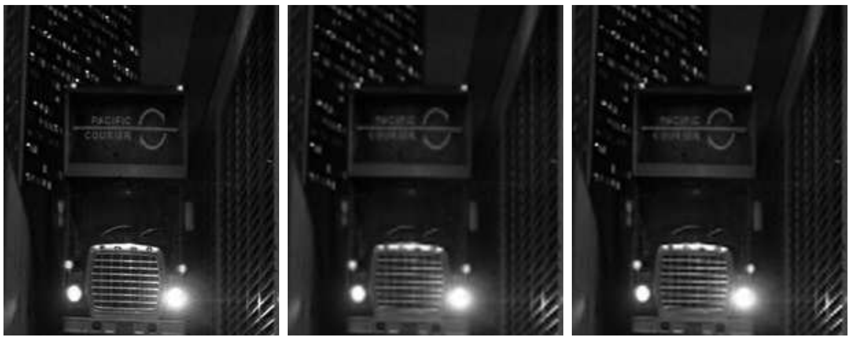
Figure 4.9: Down- and up-scaling of the sequence `Straw Hat`. (a) is downsampled with factors 0.5x0.5 as shown in (b). (c)-(e) are 2x2 (V)SR results computed from (b). Cutout sizes: (b) 45×57 pixel, the rest 89×113 pixels. Best viewed on-screen, optimally switching between the bitmap files provided in the online material [59].

4.4.8 Down and Up Again: Objective Results

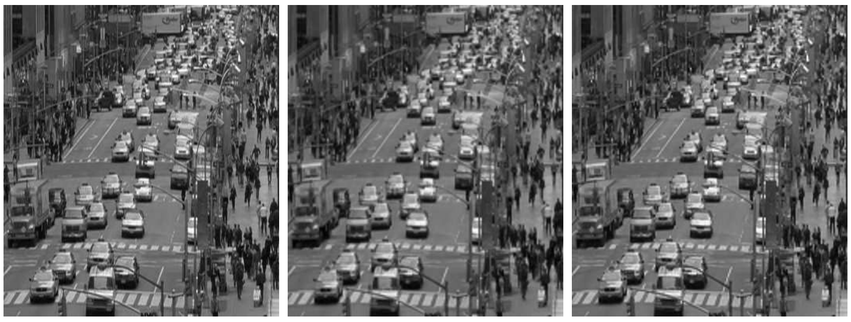
So far we have concluded that simultaneous VSR produces the highest quality outputs, but the difference to the simpler method bicubic interpolation has not been that very significant in term of sharpness when taking and average over all 19 sequences in test. In this section and the next we will show that the difference can be very significant.

A typical method for evaluation of (V)SR algorithms is to downscale an image (sequence) by the inverse of the magnification factor(s) to have a ground truth to compare upscaling results to. The downside of doing such up and down tests is that the way one downscales can effect the results. Typically a Gaussian blur kernel is used as a preprocessing step, and the chosen variance of this kernel decides where to draw the line between detail preservation and aliasing in the downsampling. One can argue that the using the Gaussian for downsampling will model the image formation process given in (4.1), but as we have discussed earlier there is hardly any blur in modern lenses and CCDs samples the signal uniformly. Thus we have chosen to just use the projection R as given in (4.12) to downscale without any pre-blurring. In the tests presented in this section we do 2x2 (V)SR, so we need to downscale with factors 0.5x0.5.

As can be seen in Figure 4.9 we are not able to recreate the original data in 4.9(a) when upscaling from 4.9(b). The results from doing bilinear and bicubic interpolation shown in Figures 4.9(c) and 4.9(d) respectively, are very smooth while the simultaneous VSR result in Figure 4.9(e) is sharper and has much more details. For this sequence `Straw Hat` and the two other sequences in test in this sequence, `Truck` and `Street`, full frame stills are given as bitmap



(a) Original (b) 2x2 bicubic (c) 2x2 simultaneous VSR



(d) Original (e) 2x2 bicubic (f) 2x2 simultaneous VSR

Figure 4.10: Down- and up-scaling of the sequences `Truck` (cutout size: 241×201 pixels) and `Street` (cutout size: 396 × 350 pixels). Best viewed on-screen, optimally switching between the bitmap files provided in the online material [59] (bilinear result also provided). Alternatively the zoomed versions given in Figures A.7, A.8 and A.9 of Appendix A can be used.

images in the online material [59]. Single frame ground truth, simultaneous VSR, bicubic and bilinear are given along with simultaneous VSR videos in the folder `Truck_StrawHat_Street_2x2_DownAndUp`.

For the two sequences `Truck` and `Street` the results are the same as with `Straw Hat`, simultaneous VSR gives much sharper and more detailed results. The original and the results of bicubic interpolation and simultaneous VSR are shown in Fig. 4.10. When switching between the bitmaps of `Truck`, notice how the lit windows in the building in the background seems to light up in the result of simultaneous VSR compared to the results of both bicubic and bilinear interpolation.

Comparing the simultaneous VSR result with the original we still lack some details and there is some blockiness on high contrast edges. The blockiness is found in all three types of upscaling, e.g. on the wall grille to the right of the truck in `Truck`. Most likely some Gaussian blur in the downsampling e.g. as a preprocessing step, or the use of a semi-Gaussian PSF like in [86] will remove the blockiness but will also give a loss of details.

As we have original ground truth sequences available, we have computed the mean square error (MSE) and the peak signal to noise ratio (PSNR) for the three sequences in test and the results are given in Table 4.1.

	Bilinear		Bicubic		Simultaneous VSR	
Sequence	MSE	*PSNR*	MSE	*PSNR*	MSE	*PSNR*
`Straw Hat`	83.03	*28.94*	73.63	*29.46*	44.72	*31.63*
`Truck`	30.56	*33.28*	24.97	*34.16*	15.16	*36.32*
`Street`	178.3	*25.62*	151.1	*26.34*	102.7	*28.02*

Table 4.1: Objective quality assessment, MSE and PSNR for the down- and upscaling experiment. Results are given for bilinear interpolation, bicubic interpolation and simultaneous VSR, the latter clearly outperforming the two others.

The MSE is

$$MSE = \frac{1}{N}\sum_{\Omega}(u - u_{gt})^2 \tag{4.28}$$

where u_{gt} is the ground truth and we sum over all N pixels of the sequence (the domain Ω).

The PSNR is

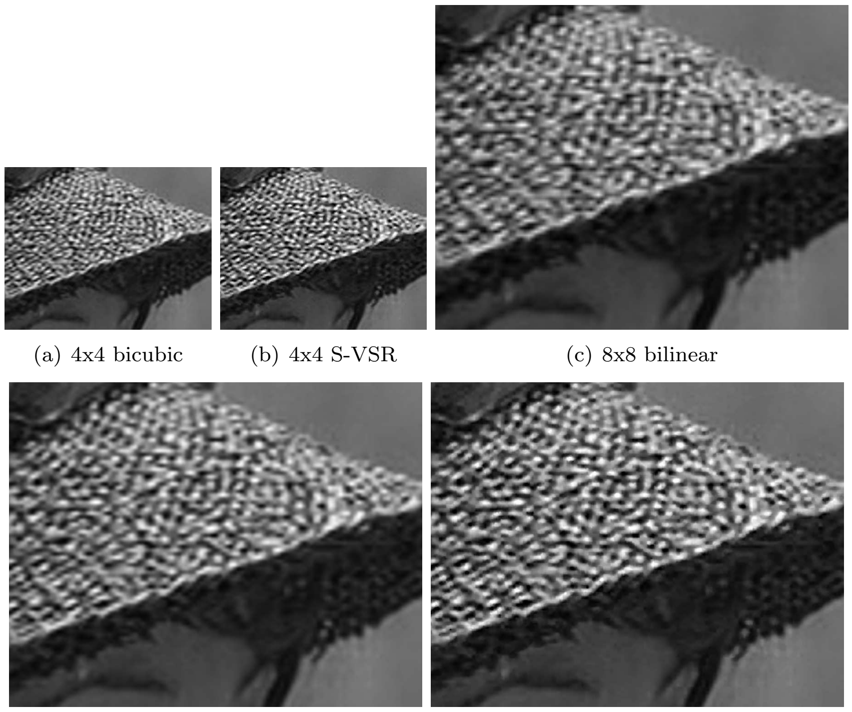
$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \tag{4.29}$$

with inverse ordering of the MSE and measured relative to the maximum possible grey value, 255.

As with the subjective evaluation, we can conclude from the MSE-values that bicubic interpolation performs better than bilinear interpolation, and that simultaneous VSR is by far the best of the three. The good correspondence between subjective evaluation and MSE is most likely due to the fact that we do not have problems with HVS-unfriendly artifacts as in the deinterlacing. The HVS-unfriendly artifacts are typically very strong but also very local giving only a small increase in the globally measured MSE. Our 'up and down' experiment is too small to draw any final conclusions but our results indicate that MSE (and thus also PSNR) are useful in evaluating (V)SR results when ground truth data is available.

4.4.9 Attempting to Break the Limits of Super Resolution

Baker and Kanade discuss the limits of super resolution in [2] and claim that it is mainly the ability of the prior to mimic or model the image content, which decides how much one can magnify: Too large magnification factors will impose too much noise in the result. To break these limits Baker and Kanade suggest using hallucination, a prior learned on specific image content types, e.g. faces or text. This gives highly detailed images at rather large magnification factors, but they do not avoid some ringing and enhancement of unwanted details, that is, noise. Using advanced, content specific learning prior on our problem of upscaling general video would require a complete and nearly perfect image content detection an segmentation system, which do not exist (yet?). We try instead to push details through the temporal consistency, but since optical flow computations on arbitrary video is still a much harder problem than e.g. registering face images that you have yourself transformed and downscaled to each other, we do not get the same detail level as seen e.g. in [2] at high magnification factors.


(a) 4x4 bicubic (b) 4x4 S-VSR (c) 8x8 bilinear

(d) 8x8 bicubic (e) 8x8 S-VSR

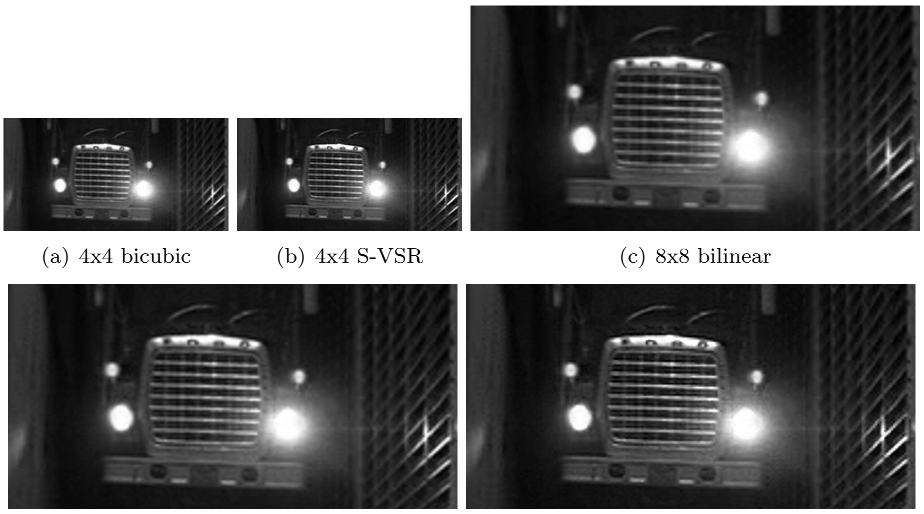
Figure 4.11: 4x4 and 8x8 VSR on `Straw Hat`. Results are best evaluated on-screen either by viewing the bitmap files given in the online material [59] (bilinear result also provided) or the zoomed versions given in Figures A.10, A.11 and A.12 of Appendix A. Image sizes are 356 $\times$ 453 pixel (4x4) and 711 $\times$ 905 pixels (8x8).

We have tested our simultaneous VSR at 4x4 and 8x8 magnification to find the limits of our algorithm and show its advantages over bicubic and bilinear interpolation. To get 4x4 and 8x8 magnification respectively we have run our algorithm with 2x2 magnification in cascade two, respectively three times. This is simply multiresolution VSR, which helps to optimize results at high magnification factors.

In Figure 4.11 showing result on the sequence `Straw Hat` it is seen clearly how simultaneous VSR performs much better than bicubic interpolation at both 4x4 and 8x8 magnification – and how bad bilinear interpolation really is (goes for 4x4 as well although it is not shown.)

As can be seen in Figure 4.11(b) and more clearly in Figure 4.11(e) the use of total variation does give a touch of the cartoon-like look typical for total variation, but it is a small price to pay compared to gain in details and sharpness over bicubic interpolation. The observations from the test on `Straw Hat` are all confirmed in the test on `Truck` as shown in Figure 4.12. For both sequences 4x4 and 8x8 bilinear, bicubic and S-VSR results are given as bitmaps in the online material [59] in the folder `Truck_StrawHat_4x4_8x8`.

In our opinion the loss in naturalness when using simultaneous VSR is small, but it is a matter of individual preferences. Doing 8x8 magnification is border-



(a) 4x4 bicubic (b) 4x4 S-VSR (c) 8x8 bilinear

(d) 8x8 bicubic (e) 8x8 S-VSR

Figure 4.12: Breaking the limits on `Truck`. Results are best evaluated on-screen either by viewing the bitmap files given in the online material [59] (bilinear result also provided) or the zoomed versions given in Figures A.13 and A.14 of Appendix A. Image sizes are 400×800 and 800×1600 pixels.

line with respect to the limits of super resolution. We do not create artifacts, but the spatial tv prior and the temporal diffusion cannot bring out sufficient detail to give a fully natural look. When looking at the 8x8 simultaneous VSR results the biggest problem is not lack of sharpness but lack of what could be called natural detailedness, and even with better priors (e.g. learning based) and/or more reliable and accurate optical flows we believe these details have to be added, they cannot be pulled out of the image at these high magnifications, they are simply nonexisting in the recordings and these high magnifications are trying to bend the subsampling theorem too far.

To finish the discussion on the limits of super resolution, we do not get the over-enhancement problems that Baker and Kanade [2] and others get. But we do not get the same level of details either. Baker and Kanade replaces the problem of noise amplification with ringing artifacts and we get the cartoon look of total variation. In both cases we lose the naturalness of the images/frames.

The limits of super resolution truly lie how much one can magnify an image (sequence) still balancing the level of noise/artifact with detail enhancement while pushing the magnification factors upwards. It is finding the right tradeoff while still pleasing the human observer with (seemingly) natural images and frames. (When the magnification is just a preprocessing step to some image (sequence) analysis, the limit is likely to be different.)

Thus breaking the limits of super resolution is not straightforward, not even with Baker and Kanade's hallucinations [2] and similar methods. Natural looking results at above 4x4 magnifications are (still?) out of reach. It is also questionable whether such large magnifications will be useful anywhere but in saving bits in coding.

4.5 Improved Video Super Resolution

We consider our simultaneous VSR algorithm as finally developed in terms of output quality for now and are focussing on implementing it as a software plug-in for editing suites and as realtime hardware (as field programmable gate-arrays, FPGAs). This is not an unrealistic goal as our algorithm is highly parallelizable, the same operations are performed on each pixel. We do not get the ideal result obtained on the simple case of the skew sequence (Fig. 4.1) in real world cases, thus we do have some ideas on how to improve the output quality in possible later versions of simultaneous VSR. First, the distribution (total variation) and filters we use in our model could be improved. As we know the eye is able to do SR, improved modelling could come from learning what filters are used in the HVS (lower level vision is already to some degree modelled by Gaussians and Gaussian derivative filter kernels). Learned priors as in [2] could also be a possibility as mentioned in the previous section. The HVS inspired priors might also be complex and computationally heavy, so we believe structure tensor based VSR to be the most likely next version – that is next after adding the gradient constancy assumption to the intensity calculations in hope of sharper edges and more details.

The gradient constancy assumption might help transport more information from neighboring frames but a higher gain in details is expected from improving the accuracy of the optical flows computed. The results produced with our old nonsimultaneous VSR from [54] are not far in quality from the results produced by our new simultaneous VSR, thus indicating that our flow might not be as precise as we could wish for. How much can be done is however an open question as most developments of optical flow is done minimizing the angular error on computer generated sequences or they are developed for highly specialized and limited application, e.g. satellite films of cloud systems or spatiotemporal medical scan data and not on real world, general video data.

4.6 Conclusion

The simultaneous VSR method presented in this chapter is in terms of output quality clearly better than bicubic and bilinear interpolation (the latter widely used in video processing devices today) and also outperforms highly expensive film post production and editing systems. There are super resolution methods found in literature that are likely to perform better than simultaneous VSR, but only on limited cases, say faces only or parametric flow only. Our method is applicable to general video with arbitrary (natural) content and motion. Real time applications of variational methods does exist today [11] and we therefore hope to see realtime S-VSR implemented in video processing systems in the near future, e.g. highly parallelized on FPGAs.

Chapter 5

Temporal Super Resolution

Temporal super resolution (TSR) is the ability to convert video from one frame rate to another and is as such a key functionality in modern video processing systems. A higher frame rate than what is recorded is desired for high frame rate displays, for video/film format conversion where also lower frame rates than recorded is requested, or for super slow-motion. In this chapter we present a novel motion compensated (MC) temporal super resolution algorithm using variational methods for both optical flow calculation and the actual new frame interpolation. The flow and intensities are calculated simultaneously in a multiresolution setting. We discuss what output quality is desired from TSR algorithms. A major problem in watching video on large and bright displays is that the motion of high contrast edges often seem jerky and thus unnatural. We test an implementation of our algorithm focussing on getting the motion of high contrast edges to seem smooth by doubling the frame rate, thus reestablishing the illusion of motion pictures.

5.1 Introduction

Temporal super resolution, full frame temporal interpolation, the conversion of a video signal from one frame rate to another is widely asked for in coding, video teleconference systems and in modern video processing systems used in production, broadcasting and viewing of television and video. In this chapter we present a novel motion compensated temporal super resolution algorithm using variational methods for simultaneous optical flow calculation and intensity calculation of the needed new frames in a sequence. We aim at application in frame rate conversions in video processors used in broadcast and home cinema systems.

Our variational TSR method is developed from a joint image sequence upscaling and restoration framework and has been used for deinterlacing (Chapter 3 of this thesis) video super resolution (Chapter 4 of this thesis) and for motion compensated inpainting by Lauze and Nielsen in [65]. In the upscaling cases, deinterlacing, video super resolution and TSR one adds information to image sequences, while one repairs damaged or missing information in inpainting and de-nosing: The framework, which is based on Bayes inference and was first used for inpainting, can also be used for image sequence de-noising if desired.

TSR is most asked for in displaying low frame rate recordings on high frame rate displays, but is also needed for both super slow-motion (super $\equiv$ high quality) and for combining different frame rate recordings into one common frame rate program. In terms of number of frames per second (fps) created from a given input TSR is not just an upscaling, but also a downscaling. In spatial super resolution (going from one discrete spatial resolution to another) upscaling is considered the difficult inverse transform of an ill-posed subsampling problem while downscaling is just the easy (well-posed) process of low pass filtering and downsampling. In upscaling you have to increase the amount of information and try to solve a subsampling problem. Upscaling in therefore what is meant with the term super resolution. So why use the term super resolution for a downscaling in time? Let us say we want to go from 25 fps to 20 fps, then we can just remove every 5th frame. The result would be usable, but the quality would be rather poor in case of motion in the sequence: every 1/4 of a second the motion would jump where a frame was skipped. To ensure a high quality result one will have to go from 25 equally space frames per second to 20 equally space frames per second. This means we will have to create entirely new frames even if we downscale in time and thus we use the term TSR also for downscaling. Creating fully new frames in time is an ill-posed problem whenever the new frames are needed sufficiently far away from existing frame. TSR is not as spatial super resolution a subsampling problem, but rather a 'wrong' sampling problem, where frames needs to be time-shifted making warping along the optical flow a natural solution. Both spatial super resolution and TSR are ill-posed *sampling* problems. Although downsampling TSR is also a hard problem, it is however a fact that with a dense sampling in time (high frame rates) the problem of downscaling becomes easier solved as the distance to nearest known neighbor decreases.

Downscaling is not what is most sought for, it is only done when needed for technical reason, like showing an American television show recorded in 60 fps NTSC in Europe where the PAL broadcast standard requires 50 fps. Upscaling in time is widely needed as most displays (projectors, plasma, LCDs and CRTs) have higher frame refresh rates than the frame rate used when recording the material. All cinematographic movies are for instance shot at 24 fps, while practically all displays today have refresh rates of at least 50 Hz. The higher display frequencies are necessary to stop the perceived image sequence from flickering. The human visual system (HVS), especially the peripheral parts of an image projected onto the retina away from the center of a focus at the fovea, is subject to flicker as these regions of the retina are highly sensitive to flickering. Thus high refresh rates are needed in displays that projects images to more than just the fovea. (Details on the HVS and a sketch of the eye are given in Section 2.2.1 of this thesis, while the projection from screen to eye is illustrated in Figure 2.2.)

Most flat panel tv-sets and most PC displays do not use real TSR but just repeat the same frame once or twice, and also in cinemas every frame is repeated two or three times to avoid flicker. Frame repetition works fine most of the time, but when the display is sufficiently bright and the viewing angle large enough (big screen tvs and cinemas with viewers sitting close to the screen) motion will start to appear jerky. The so called *phi-effect* of perceiving a sequences of still images as motion pictures is halted [73]. The problem is typically seen around high contrast edges in motion as edges are the major information perceived and

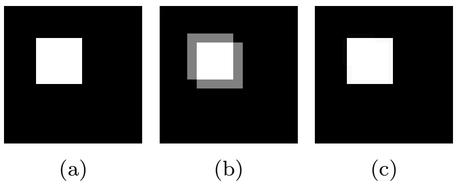


Figure 5.1: Left true frame from a sequence where the square moves diagonally, middle bad TSR by frame interpolation, and right good motion compensated TSR (variational in this case).

processed in lower level vision. The archetype example of jerky motion is a horizontal camera pan in well lit interior scenes or exterior shots of houses and cities.

Frame repetition is the simplest form of frame rate upscaling and next in complexity is frame averaging, where the two nearest known frames a weighed by the inverse of their distance to form the new frame. The results of frame averaging are double exposure-like whenever there is motion in the portrayed scene as shown in Figure 5.1(b). Better results are obtained when one computes the 2D optical flow between known frames and then compensate for the motion in the new frame. That is motion compensated temporal super resolution. Variational frameworks offer methods for computing both dense and accurate flow fields and high quality motion compensated intensity interpolation to get optimal TSR results as seen in Figure 5.1(c).

This chapter is organized as follows. In the next section (5.2) we do a problem analysis and discuss the relation between temporal super resolution and the human visual system in terms of required output quality. We also look at related work on TSR as found in literature. In Section 5.3 we describe our Bayesian based variational framework from which we derive and implement an algorithm for frame rate doubling with simultaneous optical flow and intensity calculations. The frame doubler is tested in Section 5.4. Before we conclude in Section 5.6 we discuss in Section 5.5 how to build a generic frame rate converter and the possibilities of improving the optical flows used in motion compensated variation upscaling methods (TSR, video super resolution and deinterlacing).

5.2 Background

5.2.1 Problem Analysis: Frame Rate Requirements

Temporal super resolution is done obtain a certain frame rate, as required by the receiver of the output sequence. In most cases the need is for a higher frame rate. We will focus on the frame rate requirements of humans viewers as our TSR algorithm is aimed at application in video processors in broadcast or home entertainment systems where pleasing human viewers is the final goal. The properties of the human visual system guides what minimum frame rates should be used to keep the viewing experience pleasing. The two main requirements put forth are listed below in order of importance.

- The *phi-effect* [73] should be obtained to create apparent living pictures with natural motion portrayal.

- Flickering should be avoided when displaying image sequences.

Before we look a the two requirements, we would like to stress the point that determining the exact minimum required frame rate of image sequences is a difficult, multiparameter problem. The applied display technology (film projectors in cinemas, LCD tv set, etc.), screen size, screen brightness, contrast and color reproduction, viewing distance and angle, lightning conditions in the viewing room, image sequence content, motion in the scene depicted, and aperture time of cameras, are but the main factors in the equation of defining the minimum frame rate required.

There are however some general (standardized) agreements of which frame rates to use in different media to meet the three requirements posed above. Since sometime around the 1920s almost all film recordings have been done in 24 fps [7]. The American and Asian tv standard NTSC requires 60 interlaced fields per second (also abbreviated fps), while the European PAL requires 50 fields per second. There was also a growing consensus up through the 1990's that 50 fps was no longer enough to avoid flickering as the size and brightness PAL CRT tv-sets grew too large. This gave birth to the 100 Hz technology, an example of early TSR (see for instance [32]).

The phi-effect was already discussed in Chapter 2 of this thesis, but to recap it is the effect of showing a set of still images, recorded and shown so fast after each other, that any motion in the depicted scene appears real and natural. To create the illusion of motion pictures, each frame cannot be exposed to the eye for too long as the human visual system will interpolate the simplest (linear) motion between the frames erasing any complexity in it [73]. Thus the frames need to be recorded at a rather high rate to obtain the phi-effect – at least when motion is complex.

Flickering occurs on when an image is not updated often enough, that is the update frequency (frame refresh rate) is so low that the eye senses flicker. Thus in cinemas a rotating shutter blinds out each frame shortly once or twice to get 48 or 72 Hz refresh rates. Since the eye is very sensitive to changes in the light it is exposed to – especially in the periphery of the retina away from the fovea – and since cinema screens and many larger television screens expose a large part of the eye, the flicker can be sensed at frame rates higher than what is traditionally required to obtain the phi-effect. Although one might not consciously sense any flicker it is still sensed subconsciously tiring the visual system.[1] There seems to be a consensus that refresh rates somewhere around 70 to 100 HZ will suffice to stop most flickering on most displays, but it ultimately depends on the tracking done by the eye of the human observer.[2]

[1]By looking at something next to a CRT screen without really focussing on it, the screen will now be projected to the more flicker sensitive parts of the retina and if the refresh rate is set to 60 Hz or less one will clearly see the flicker normally only sensed subconsciously as the HVS is busy with what is in focus at the fovea.

[2]While film projectors needs to change frames and CRTs has the problem of the light emission of the phosphor used on the screen drops over time, LCD monitors are not subject to forced blackouts between frames: they are hold-type image displays, not impulse-type image displays see [20]. Also plasma displays have some memory in the plasma gas to partially prevent flicker problems. Even with the flicker problem (almost) eliminated, LCDs and plasma are still subject to the need of creating the phi-effect when displaying video and thus needs higher frames rates when displaying motion.

As our discussion of the two requirements have shown, the major reason why we do temporal super resolution is to obtain the phi-effect. Flicker can be handled by simple frame repetition, but since the eye is able to track motion to a certain degree, the frequency used for sampling the signal is too low to establish the phi-effect and the motion seems unnatural: As the eye track the motion, the sampling frequency is too low and the eye senses flickering and no natural motion between the two samples can be inferred by the HVS to get the phi-effect. Thus our two requirements are closely related, with the difficult task being to prevent flicker in tracked motions to reestablish the phi-effect.

The problem of unnatural motion is most prominent when the boundary of an object moving (or some internal edge of the object) is of high contrast (large gradient). If the recorded frame rate is too low, the phi-effect is not obtained and the viewer is left with an sensation of a jerky, unnatural motion. Along with the contrast of the edge the velocity of the motion can break the phi-effect. The faster an object (or the camera) moves the more jerky the apparent motion becomes until the eye is no longer able to track the motion.

Decreasing the apparent jerkiness of fast moving, high contrast edges will be our focus when developing and testing our motion compensated temporal super resolution algorithm.

An effect that might cover up the jerky motion appearing when using too low frame rates, is motion blur. Motion blur occurs (in regions of motion) when the camera used has a long aperture time and the input in each point or pixel of the frame is integrated over a relatively long time interval.

5.2.2 Blur Acceptance in Human Vision

Before looking at related work on TSR algorithms and discussing its relevance to our work, we would like to get a more specific idea of what output quality to aim at. What is required to satisfy the final judge: the human visual system. We have discussed the HVS in Chapter 2 and would also like to refer the reader to the book [73] on human sensation and perception by Matlin and Foley, while books like [4], [82] and [107] give some insight on the adaption and accommodation of video engineering (processing, coding, transmission, displaying etc.) to the HVS.

Doing TSR in a wrong or incomplete way will most likely create artifacts in the new frames interpolated. Using a motion compensated variational method, blur will be the most likely artifact. Thus we need to know (if possible) how much blur is acceptable to the human visual system.

Judging generally and objectively how unsharp we can allow parts (say every other frame) of an image sequence to be is still an open question. Vision research does not offer an answer as the border between sharp and blurred in human perception is still sought for in more limited subproblems. The paper [22] by Ciuffreda *et al.* is a typical example. Ciuffreda *et al.* tries to find out when blur becomes bothersome on simple stationary text and letters (similar to the Snellen eye charts found at opticians etc.). With the general mapping of blur sensitivity and acceptance in the HVS still to be found, we will have to subjectively evaluate if we produces satisfying results when doing temporal super resolution.

In video and image processing (and analysis) focus is on either removing or adding blur from ones data. Removing it is done in a) de-blurring to remove motion blur, in b) super resolution to compensate for lack of resolution in the image recording and c) in de-noising as noise is normally removed at the prize of added blur. Adding blur deliberately is mostly done to remove artifacts, typically blocking artifacts from too hard coding compression, but has also been used to simulate film aperture blur in video recordings (see for instance [102] by Tschumperlé and Besserer). When processing video one wishes to know how much blur one can add or leave in the image (sequence) without annoying the HVS, which is hard to tell as objective factors like video content (local and global motion, color, level of details etc.), screens (brightness, viewing angle, resolution etc.) and physical viewing conditions (light mainly) play a major part in such an evaluation – not to mention the still immeasurable human psyche, the overall subjective judge affected by all these objective factors.

Even though we do not get an answer to our question on blur acceptance from vision research, we do get helping pointers. Doing TSR on non-moving sequences is easy, one can just repeat frames, we know it is in regions of motion that the task of doing quality TSR gets difficult. In [15] (Burr and Morgan) and [74] (Morgan and Benton) it is shown experimentally that moving objects often appear sharp to the HVS, not because some mechanism removes blur, but because the HVS is unable to decide whether the object is really sharp or not. (Try for instance to watch the video `BoatTSR50fps.avi` given in the online material [58] and see how sharp it looks as video compared to the relative unsharpness when viewing it as still in Figure 5.6.) At what motion magnitudes this effect of perceived sharpness starts to appear is unclear – when does the eye stop tracking etc. – but it seems we can allow for some blur when doing temporal super resolution and still get good results subjectively evaluated. In [20] Chen *et al.* shows that motion blur in LCD displays can be reduced by inserting blurred frames between frames that are enhanced correspondingly in the high frequencies. This indicates, that even without enhancing any frames, we can allow for blur in some frames and still hope for a total perceived sharpness. For now we will just have to try to get our new frames in the output as sharp as possible to ensure optimal results.

5.2.3 Strategies of Creating New Frames

There are three different methods for creating new frames in an image sequence:

- Frame repetition.
- Frame averaging.
- Motion compensated interpolation.

Of these three methods, frame repetition is the simplest, it solves the flickering problem and does not create any artifacts from bad temporal interpolation, but the motion portrayal stays unnatural. It is very cheap to do and any frame rate can achieved, one just have to repeat the frame recorded closest to where the current output frame is needed. When conversion ratio it not integer, e.g. 24 to 30 fps, frames will be repeated a different number of time adding some nonlinear jumps to the motion, possibly increasing the unnaturalness of the motion.

Frame repetition and frame averaging will give perfect results on sequences with no motion, but in case of motion frame averaging will give a blending/fading or superimposed effect between the two frames used in the averaging.

This will to a certain extent smooth the flow but the blending is definitely to be considered an undesired artifact as seen in Figure 5.1.

Taking motion in the depicted scene into consideration one can do motion compensated interpolations, which is what we denote temporal super resolution. Motion compensation is an essential component in TSR, non-MC does not exist in our terminology. Only when knowing the flow one can truly create the data in moving regions of the frame.

The traditional approach to temporal super resolution is to first compute the flow of the sequence and then interpolate the intensity values in the new frames. The simplest form of TSR uses linear interpolation along the flow trajectories, weighing each of the two original input frame contribution inversely by their distance to the frame being interpolated. Simple TSR gives perfect results if the computed flow field is reliable and precise, but this is rarely the case. Thus a fall back option is needed.

The flow can be unreliable or imprecise for many reasons, e.g. at motion boundaries (especially if one uses block matching), at occlusions, or because the flows in the new frames has been interpolated from flows computed on the old frames. More advanced nonlinear interpolations can be employed to handle the problems as we will discuss further when looking at related work in section 5.2.7. If the flow is just unreliable in one direction, one can give more weight to the interpolation in the other direction, but often a fall back to simple non-MC averaging is necessary (4-42% fall back is reported by Dane and Nguyen in [27]).

When interpolating or warping the flow computed between two known frame into any new frame(s) positioned between them, it will not always be so that there is a flow vector in all pixel positions of the new frame(s). The fall back described above could be used in such a case, but one could also fill in neighboring flow vectors hoping they will be correct – if not the fall back mentioned above is necessary. Exactly how complex this filling in of missing flow vectors is, can be seen in the patent [85] by Robert. Without knowing the intensities of the new frame(s), it is impossible to know if the guessed flow is correct, but to get the intensities we need to know the flow! This case of two unknown each depending on the other is truly a hen-egg problem. Typically (as will become clear in section 5.2.7) one first computes the flow from the neighboring frame and interpolate it into the new frame, which is apparently considered trivial as none of the references we have found gives any details on how the warp/interpolation is done (the patent by Robert [85] being the exception). After interpolating the flow, the intensities are interpolated.

5.2.4 Simultaneous Flow and Intensity Calculations

To truly get a chance of computing the optimal estimates of the flows and intensities one has to compute them simultaneously as we will do. The simultaneousness can be achieved by iteratively improving always using the latest version of the other. Multiresolution processing of images and image sequences creating a pyramid of spatially downscaled version of the image (or frames) is a well-established tool and is usually used for variational optical flow computations. We will use in temporal super resolution to switch between flow and intensity calculation: The flow calculation will draw on intensity data from the level above and the intensity calculation will have the flow from the current level available. Making the steps in the pyramid relatively small one will always have a highly reliable version of the intensities available for flow computations thus optimizing the following intensity computations at that level (and the flow calculations at the next level etc.). Starting with a very coarse version of the sequence at the top of the multiresolution pyramid, we initialize using just simple frame averaging as there is hardly any motion due to the downscaling. Using multiresolution we also have good spatial estimates of intensity values available at any given position in the new frames allowing us to have a spatial fall back to replace the temporal fall back when flows are unreliable. Using spatial fall back has to our knowledge not been done before. The temporal fall back to frame averaging is really a bad solution whenever there is motion in a sequence, but spatial fall back can also be really bad, if there is large regions of unreliable or missing flow, but this is rarely the case when using variational flow computations as compared to block matching. Variational flow algorithms compute dense and highly accurate flow fields (see for instance the survey by Bruhn *et al.* in [12]). Should the flow be unreliable, then spatial fill-in using a good spatial image model in multiresolution settings (as we will do) will provide estimates of the image content that are realistic to the human visual system and most certainly a lot better than some blended output resulting from frame averaging.

5.2.5 Worst and Best Case: Frame Doubling

Frame doubling is the most difficult case of temporal super resolution. Exactly in the middle between two known frame we have the maximum distance to known frames, and thus it is the position where any method will have the highest uncertainty in its prediction of the new frames. Test results showing this can be found in [78] by Ojo and de Haan.

Since we can keep all original frames in frame doubling and since no other frame rate conversion will allow for 50% original data in the output (under the logical requirement of equally space frames) frame doubling is the easiest case of TSR – given we do not expect new frame to be as good as the old frames in all cases.

Considering the two criteria given above on how to decide worst/best cases, we believe the challenge lies in creating sharp and artifact-free frames anywhere not close to an original frame. Thus the real worst case is any frame rate conversion where we need longer parts of the output to be purely new frames positioned relatively far from original frames. This happens when we only change the frame rate a little, e.g. from 24 fps film recordings to 23.98 fps (NTSC progressive rate) where we will have long periods with new frames close to original frames alternating with long periods with new frames far away original frames. These small changes in frame rates are often done by altering the speed of the recording, e.g. 24 fps film is speed up 4% to be shown in 25 fps progressive PAL. (Each frame is split into two fields to get 50 fields per second interlaced). However, true changes of the frame rate without changing the real time speed is sought for in the broadcasting industry.

5.2.6 Super Slow-motion

As we have discussed in section 5.2.2, the requirements on how sharp the new frames need to be is not clear, we might make do with somewhat blurred frames.

The sharpness we end up with is highly depending on the quality of the flow. We will test using two types of optical flow, one which is 100% theoretically consistent with how we formulate our framework of joint flow and intensity calculation, and another which is inconsistent and takes longer to compute, but also gives very sharp, well-segmented flows. Shaper flow is expected to give sharper new frames.

Super slow-motion is parallel problem to frame rate conversion, but since the temporal super resolution out when used for super slow is viewing at lower frame rates (same as the input was recorded in, but stretched in time) the need for very sharp frames is much more explicit than in frame rate conversion, thus one will likely need shaper new frames.

5.2.7 Other Work

Temporal interpolation of signals is not new, it has been done for a long time for 1D signals in signal processing, but these methods cannot be applied to our problem as we have motion to consider. If we compute the flow of a sequence it should just be possible to take a simple average along the backward and forward flows to create new frames. This is however an oversimplification of the problem as we do not know what comes first in a new frame: the flow or the intensities? To optimally solve this hen-egg problem one has to iterate between estimating the flow and the intensities as we suggested above. This idea of simultaneous multiresolution flow and intensity calculation has already been proven itself useful for motion compensated inpainting in [65]. Let us have a look at what others have done before us in the field of temporal super resolution.

In medical imaging interpolation of new frames or volumes of a time sequence of 2D or 3D scans are of interest, mainly in lung (respiratory gated) and heart (heart gated) imaging. The work by Ehrhardt *et al.* in [35] is a typical and recent example, where temporal super resolution in heart gated imaging is performed using an accurate flow algorithm, but doing simple motion compensated interpolation of intensities along the flow lines to get the new frames. In our own field of video processing there are several TSR patents like the one by Cornog *et al.* [24] and the already mentioned [85] where the same procedure as in [35] is used: Flow calculation (good or bad) followed by some non-iterative averaging along the flow. TSR is also done in integrated circuits (ICs) as described by de Haan in [28] using 8×8 block matching flow with a median filter for motion compensated interpolation (details on the intensity interpolations is given in [78] by Ojo and de Haan). In a recent paper [27] by Dane and Nguyen motion compensated interpolation with adaptive weighing to minimize the error from imprecise or unreliable flow is presented, which is surely needed as the flow used in [27] is the MPEG coding vectors (typically prediction error minimizing vectors). The advantage of using MPEG vectors as flow is that one avoids the computationally expensive flow calculation.

In [52] Karim *et al.* focus on improving block matching flow estimation for motion compensated interpolation in low frame rate video and no less then 16 references to other TSR algorithms are given. An overview of early work on motion compensated temporal interpolation in general (TSR, coding, deinterlacing etc.) is given by Dubois and Konrad in [33] where they state that even though motion trajectories are often nonlinear, accelerated and complex, a simple linear flow model will suffice in many cases. In [17] Chahine and Konrad shows that modelling the acceleration will improve results in motion compensated TSR when measuring the picture signal to noise ratio (PSNR) between the results and the ground truth. The improved flow modelling will make interpolation (and prediction) coding better in terms of quality to bandwidth ratio. We are not necessarily interested in optimizing an objective error measure like the PSNR, but are more focussed on pleasing the human viewer. Variational optical flow algorithms can model and calculate acceleration as a consequence of temporal regularization. An example of this is the flow of the sequence `Ettlinger Tor` computed by Brox *et al.* in [9], but in interpolating flow and intensities in all new frames, a straight motion trajectory is likely to be chosen as the flow of minimum energy (variational optical flows are typically found by minimizing the energy of the model applied to the image sequence and its optical flow). Specifically modelling acceleration in the flow will give a smoother flow trajectory in TSR. Whether this increased smoothness of the flow will improve the quality of TSR is doubtful as the human visual system itself does linear interpolation.

A problem somewhat more complex than our new frame interpolation problem is trying to create a new arbitrary viewpoint 2D sequence from a multicamera recording of a scene as done by Vedula *et al.* in [105]. The 3D registration and scene modelling (similar to structure from motion problems) is what complicates matters, but the multicamera recordings does at the same time provide you with an abundance of available information. Leaving out all the 3+1D shape and flow modelling, the 2+1D TSR part used in [105] is the classic Lucas-Kanade flow estimation [71] followed by simple motion compensated intensity interpolation using weighing by linear distance in time from forward and backward known frame neighbors. No optical flow estimation is performed by Shechtman *et al.* in [92] and in their version of the multiple camera approach, all the cameras are assumed to be close spatially or the scene assumed planar to allow the different input sequences to be registered by simple alignment to the common frame of reference. From the multiple inputs a high resolution output in either space or time – it is a tradeoff – is produced. Shechtman *et al.* therefore already have a very dense recording of a registered scene making it very easy (according to the authors) to produce high frame rate TSR outputs without flow estimation (e.g. 75 fps from four 25 fps sequences). This technique can not be used on single camera recordings and the view point can no longer be chosen arbitrarily as it could in [105].

Using patches to represent salient image information is well-known (see e.g. the papers by Freeman *et al.* [40] and Griffin and Lillholm [44]) and an extension to spatiotemporal image sequences under the name of 'video epitomes' is presented and used for TSR by Cheung *et al.* in [21]. The framework for video epitomes can just as our Bayesian inference based framework be used in general for image sequence inpainting, upscaling and de-noising. In the case of TSR it is not at all discussed in [21] to which degree video epitome TSR can handle motion and in the example given on generating frames dropped unevenly in an Internet broadcast there is only very little motion. Furthermore video epitomes need to be learned on either a better representation of the degraded data (e.g. high resolution sequences) or on the input data available. Learning is a computationally very costly process (no details given in [21]) and it is therefore unclear whether video epitome TSR can be applied to general video at all (and at what processing cost in time and hardware) but video epitomes is as such an interesting technology.

5.2.8 Benchmarking in Testing

When presenting a new solution to a problem any claims of improved results or performance should be supported by a reasonable validation of the claims. In medical imaging requirements on validation are obviously strict and often very formalized, whereas they are less formalized in image processing where one 'just' enhances data.

In evaluating enhancement of images or image sequences, e.g. in TSR, the choice is between objective and subjective quality assessment as discussed in Section 2.3 of this thesis. These quality measures are then used to show the difference in quality between the suggested new method and other methods. (To use objective measure a ground truth version of the sequence needs to be available) This is exactly as we have done for deinterlacing in Chapter 3 of this thesis and for video super resolution in Chapter 4 of this thesis. In most cases the methods used in comparison are rather simple as you often has to balance the effort between improving your own method and implementing other advanced methods.

In [52] Karim *et al.* test several TSR algorithms, the difference between the algorithms being the specific block matching algorithms used in the flow calculations. They all use simple averaging along the flow as intensity calculation, thus making it a study on the quality of block matching when used in motion compensated schemes (in this case TSR). In [27] Dane and Nguyen focus on testing different weighing schemes for the intensity interpolation and simple averaging along the flows is compared to two adaptive weighing schemes, the adaption being based on either reliability of the flow vectors or the prediction error (as in video coding). To directly compare the performance of our variational TSR algorithm to the performance of any of the above, we would have to test our method on the same data as they have used in their tests.

As it is now, authors rarely use the same input data to produce their results (as given in papers). There do exist a number of sequences commonly used for video processing (coding primarily) including sequences as Flowers, Stefan, Foreman, Carphone, Mobile calendar (mobcal), Bicycle, Tokyo, Football etc., which has been used in TSR testing. Unfortunately different subsets of this data set is used (selected sequences or frames and at different resolutions, QCIF, SIF, CIF etc.). Even with partial overlaps between different TSR tests, visual results given as figures in the papers (video results are rarely given) are too small to really compare, or different frames are selected for display. Since we cannot find comparable results in literature, we could then implement the methods of others, but as mentioned above, it often takes resources away from developing your own method – unless what you do is based on earlier methods. Since basically all TSR methods we have come across use block matching, we would have to build up an expertise in that field, but have not (yet) done so as it is not done easily.

Furthermore limits on paper lengths often prevents the full story with all details in an algorithm to be given and thus (extensive) dialog with authors is also necessary.

Next option to get directly comparable results is to ask authors to run their code on the same data that we use in our tests or to ask for a copy of their code (if the system exists as software for standard PCs). This can be troublesome due to (research) political issues, especially if any commercial interests are involved.

But some labs do share their code, an example being the spatial video super resolution and demosaicing work by Farsiu *et al.* presented in [37] and [38].

The ways of obtaining directly comparable results we have discussed so far have all been troublesome, but there is one last and mutually beneficial way to go, which is to develop a joint taxonomy for benchmarking in TSR (or any other field of application). An example of this is the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, where the tenth is held in the fall of 2007. Here all algorithms presented in workshop papers are to test on the benchmark data given by the workshop organization (see `http://pets2007.net`). It is in no way guaranteed that work on temporal super resolution can carry a series of workshops, but just one founding workshop or meeting agreeing on a benchmark data set would be enough.

A risk when forming a benchmark is that the test set might end up being too big, leading to selective testing which is no better than what is already done. On the other hand the test set should not be too small as it should both deal with the major problems in TSR and be representative of video as such: We cannot have a TSR method solving the hard cases but creating artifacts in other types of video. We suggest a few examples of unnatural motion (mainly high contrast edges in sequences with camera motion given a jerky motion effect) and some example of complex motion to see that the motion estimation can handle more than just translational motion.

But even with a data set for benchmarking selected and generally agreed upon, there still is the question of how to evaluate the results, should it be done objectively and/or subjectively and what exact method(s) of evaluation should be used?

5.3 Theory and Algorithm

The temporal super resolution algorithm presented here was derived from a variational framework based on a Bayesian inference for restoring and enhancing image sequences as presented in Section 2.6 of this thesis. We recapitulate the framework here and give its specific use in TSR before we describing the two versions of a frame doubling algorithm we have implemented.

5.3.1 Bayesian Inference and Variational Framework

Probability distributions are a good starting point for modelling image sequences and their optical flow. In its general form for adding information to an image sequence it is

$$p(u,\vec{v}|u_0,D) \propto \underbrace{p(u_0|u,D)}_{P_0}\;\underbrace{p(u_s)}_{P_1}\;\underbrace{p(u_t|u_s,\vec{v})}_{P_2}\;\underbrace{p(\vec{v})}_{P_3}. \tag{5.1}$$

where $\vec{v}$ is the optical flow of the output sequence u, and u_0 is the input sequence. u_s and u_t are the spatial and temporal distribution of intensities respectively. D is the domain where data should be added, the holes in inpainting and the set of the new frames in TSR. On the left hand side we have the a posteriori distribution from which we wish to extract a maximum a posteriori (MAP) estimate. The right side terms are: P_0, the image sequence likelihood, P_1 the spatial prior on image sequences, P_3 the prior on motion fields, and P_2 a term that acts both as likelihood term for the motion field and as spatiotemporal prior on the image sequence. We use the Bayesian to variational rationale by Mumford [75], $E(x) = -\log p(x)$ to get to a variational formulation. We are then given a continuous energy minimization problem of the form

$$E(u, \vec{v}) = E_0(u, u_0) + E_1(u_s) + E_2(u_s, u_t, \vec{v}) + E_3(\vec{v}). \tag{5.2}$$

Under mild regularity assumptions, a minimizing pair $(u, \vec{v})$ must satisfy the condition $\nabla E(u, \vec{v}) = 0$ where ∇ is the gradient and the solution expressed by the coupled system of equations

$$\begin{cases} \dfrac{\partial E}{\partial u}(u, \vec{v}) & = 0 \\ \\ \dfrac{\partial E}{\partial \vec{v}}(u, \vec{v}) & = 0. \end{cases} \tag{5.3}$$

This system can, depending on the level of integration in the actual implementation and numerical solution chosen, be considered simultaneous. The simultaneousness comes from alternatingly updating the guesses on solutions to $\partial E/\partial u = 0$ and $\partial E/\partial \vec{v} = 0$ down through a multiresolution pyramid as discussed in section 5.2.4. We thus minimize both the flow and intensity energy on each level in the pyramid as we iterate down through it. Multiresolution is a well-known technique from pure optical flow calculations (see for instance Brox *et al.* [9]). We will get back to the use of multiresolution in temporal super resolution in Section 5.3.6.

5.3.2 Variational Temporal Super Resolution

Returning to Equation (5.2) we need to define the functions of each of the energy terms.

First, E_0 would control the strength of diffusion in de-noising, but in TSR it to defines where the frames of the output u is located and how it relates to the input data u_0. It tells us to keep original intensity data u_0 untouched as it represents our anchor points for creating new data and should not be altered.

The most important term is where E_2 we want to temporally transport information into the new frames along the flows (forward and backward). It acts both as a prior on the intensities and as likelihood on the flow modelling the consistency of the intensity data along the flows, it is the brightness constancy assumption (BCA). WE use the linearized version of the BCA, the optical flow constraint (OFC) dating back to the pioneering work of Horn and Schunck [48]. (The BCA and the OFC are discussed in Section 2.5 and definitions given in Equations (2.1) and (2.3) respectively.)

Horn and Schunck also defined the need for a prior on the flow, which in (5.2) is represented by the E_3-term. It serves the purpose of filling in good estimates of flow vectors in smooth regions from accurate values calculated where salient image data is available (edges, corners etc. giving nonzero image gradients). It also give reasonable guesses for flow vector in occluded areas, but these guesses are mostly ignored by the intensity calculations at we have both forward and backward flows available.

E_1 is the classic, spatial regularization term and will in case of unreliable flow give spatial filling in. To keep it from doing (inaccurate) spatial diffusion where flows are reliable and temporal information thus correct, detailed and sharp, it should be given very small weight. The control gained by explicitly giving the E_1-term a small weight is and augmentation of the implicit control of our algorithm. Giving a low weight to the spatial diffusion term will jut tell the algorithm to only start using spatial information when the temporal information is really bad due to imprecise or unreliable flows.

The distribution of each of the energy terms E_i ($i = 1, 2, 3$) in (5.2) are in standard variational image and video processing of today either a Gaussian or total variation. Both allow smooth regions while total variation also allows for edges in its modelling of images and image sequences. The edges can be both standard spatial edges in each frame of the sequence and temporal edges in the sequence as caused by (dis)occlusions. In minimizing the intensity energy (solving $\partial E(u,\vec{v})/\partial u = 0$) total variation on E_2 will stop diffusion in the direction of unreliable flows (e.g. at occlusions) as the incorrect flow forms a temporal edge. When we need to resort to spatial diffusion in the intensity data, total variation on E_1 will make sure edges are preserved and not smoothed out – as they would be if we used the Gaussian. On E_3 total variation will also preserve edges in the flow (boundaries segmenting the flow). Because of the edge preserving properties of the nonlinear total variation we chose it over the linear Gaussian although it comes with a price: The numerical implementation is more complicated and the running times longer, but these negative effects are outweighed by the increase in output quality. With the use of total variation and the E_i's ($i = 1, 2, 3, 4$) discussed above, (5.2) becomes

$$\begin{aligned} E(u,\vec{v}) \quad = \quad & \underbrace{\lambda_1 \int_\Omega \psi(|\nabla u|^2)dx}_{E_1} + \underbrace{\lambda_2 \int_\Omega \psi(\left|\pounds_{\vec{V}} u\right|^2)dx}_{E_2} + \\ & \underbrace{\lambda_3 \int_\Omega \big(\psi(|\nabla_3 v_1|^2) + \psi(|\nabla_3 v_2|^2)\big)dx}_{E_3}, \qquad \underbrace{u = u_0|_K}_{E_0} \end{aligned} \tag{5.4}$$

where $\nabla = (\partial x, \partial y)^T$ is the spatial gradient, $\nabla_3 = (\partial x, \partial y, \partial t)^T$ is the local spatiotemporal gradient, and the λ's are positive constants weighing the terms with respect to each other. x runs over the whole image sequence domain Ω and K is the domain of known data, where K may overlap the domain of the output sequence C, e.g. in frame doubling where it is composed 50/50 of D and K (for frame doubling $C = D \cup K$) and thus we need E_0 to be $u = u_0|_K$.[3] v_1 and v_2 are the x- and y-components of the flow field, i.e. $\vec{v} = (v_1, v_2)^T$ and $\vec{V} = (v_1, v_2, 1)^T$). As discussed in Chapters 3 and 4 $\pounds_{\vec{V}} u$ denotes the temporal Lie-derivative of u along the flow $\vec{V}$ (the $\vec{V}$-directional derivative of u) and is the optical flow constraint. $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is an approximation of the $|\cdot|$ function as the latter is non- differentiable at the origin. ε is a small positive constant (10^{-8} in our implementation).

Splitting the energy (5.4) according to (5.3) we get this energy to be mini-

[3]Since we allow for both up- and downconversions of frame rates, Ω can be either equal to C or K depending on which is longest in time.

mized for the intensities

$$E(u) = \underbrace{\lambda_s \int_\Omega \psi(|\nabla u|^2)dx}_{E_1} + \underbrace{\lambda_t \int_\Omega \psi(\left|\pounds_{\vec{V}} u\right|^2)dx}_{E_2}, \quad \underbrace{u = u_0|_K}_{E_0} \tag{5.5}$$

where $\lambda_s = \lambda_1$ and $\lambda_t = \lambda_2$ in (5.4). For the flow we need to minimize

$$E(\vec{v}) = \underbrace{\lambda_2 \int_\Omega \psi(\left|\pounds_{\vec{V}} u\right|^2)dx}_{E_2} + \underbrace{\lambda_3 \int_\Omega \big(\psi(|\nabla v_1|^2) + \psi(|\nabla v_2|^2)\big)dx}_{E_3}. \tag{5.6}$$

In order to improve quality, the brightness constancy assumption in E_2 can be supplemented with the gradient constancy assumption (GCA) (see section 2.5 of this thesis) as it is done in the inpainting algorithm by Lauze and Nielsen in [65]. Using the GCA in the intensity minimization adds a serious layer of complexity, while it more unclear to what degree it will improve the quality of the resulting output image sequence. The use of the GCA in $E(u)$ is discussed in Section 2.6 and Chapters 3 and 4 of this thesis.

5.3.3 Gradient Constancy Assumption on the Flow

In the temporal super resolution literature (e.g. [17], [24], [27], [28], [52], [78] and [85]) the flow estimation is considered the important and difficult part of any TSR algorithm as advanced intensity interpolations are only suggested to remedy errors in the flow. But even the advanced intensity interpolations are far from as complex as the flow computations. Flow computations are in general more complex and the need for segmentation (which is one of the things the GCA should helps do) is more explicit in the flow – if the flow in a TSR algorithm is well-segmented, then the intensities will be as well.

From the work on variational optical flow by Brox *et al.* in [9] and the survey of optical flow methods (practically a benchmark) by Bruhn *et al.* in [12] it is clear that the gradient constancy assumption will improve the precision and reliability of the flow. The formulation of our TSR algorithm has so far been consistent with the idea of joint flow and intensity optimization and we are theoretically minimizing the joint energy of (5.2). In theory a 100% simultaneous solution will minimize (5.2) directly to solve $\nabla E(u, \vec{v}) = 0$, but in practice we are minimizing the split energies in (5.5) and (5.6) according to (5.3). Thus we can add terms to either (5.5) or (5.6) independently and we have chose add the (linearized) GCA to the flow energy. E_2 with GCA (flow energy only) then becomes

$$E_2(\vec{v}) = \int_\Omega \Big(\lambda_2 \psi(\left|\pounds_{\vec{V}} u\right|^2) + \gamma \psi(\left|\pounds_{\vec{V}} \nabla u\right|^2)\Big)dx \tag{5.7}$$

where γ is a positive, constant weight and $\pounds_{\vec{V}} \nabla u$ the $\vec{V}$-directional Lie-derivative of ∇u, that is the linearized GCA. If we set $\gamma = 0$ in our implementation, we are back at (5.5) and thus we are able to test both with and without GCA in one joint implementation.

Even if we should be able to estimate a perfect flows one day, simple interpolation it not always enough. Whenever there is non-integer motion we need spatial subpixel interpolation. In TSR we have a hen-egg problem making a perfect flow impossible to obtain. Since the TSR problem is ill-posed and solutions to it thus suboptimal, it would be interesting to see if using the GCA in the intensity energy $E(u)$ will improve variational TSR results.

5.3.4 Temporal Super Resolution Is not Inpainting

Inpainting small holes in a sequence is very different from creating (several) full new frames of a sequence. Claiming that inpainting and TSR are the same is the same as saying de-noising and inpainting are the same: It is all true in a broad scene as shown with our Bayesian framework (or the video epitome formulation by Cheung *et al.* in [21]) where the difference between the applications lies in the formulation of the data term P_0 (and equivalently E_0). In de-nosing it is simply a weight on how hard to regularize all data, in video super resolution it is the somewhat more complex super resolution constraint governing the projection of data back and forth between low resolution and high resolution frame (see Chapter 4). Solving any of the problems, de-noising, inpainting, deinterlacing, video super resolution or temporal super resolution in practice, shows that the similarity is mainly theoretic and lies in the common modelling of image sequences content. Solving the actual problems are very different.

The major difference lies in the support or masking of the problems. In de-noising all data is available for support, but then again any of the pixels might be noisy. In inpainting the noise can be said to have grown in size, but in contrary to de-nosing the damaged regions are located manually or automatically before inpainting. On the downside, inpainting only has spatial support from the boundaries of the damaged regions, which is problematic if the damages are large. Still image inpainting relies 100% on spatial information support, whereas image sequence inpainting also has temporal support from (mostly undamaged) neighboring frames. In temporal super resolution we only have temporal support and cannot rely on any spatial support to help solve the problem of missing information. Smaller holes in image sequences that are inpainting reasonably well are also easier overlooked by the human observer than full frames created reasonably well as full frames takes up more visual attention than only a small parts of frames. As image sequence modelling improves in the future it also likely that TSR will benefit more from it than inpainting – depending on the hole sizes of course: At 100% of the frame size inpainting then is TSR.

5.3.5 Motion Blur

Some would claim that motion blur should be included in modelling temporal super resolution in the sense that TSR aims at sharp sequences. We (dis)respectfully disagree. We model motion blur in the sense that we preserve it. If a film maker or tv-producer has created motion blur, it is part of the artistic expression and should be preserved. We consider the input frames as perfect and want to get the new frame to ultimately resemble them as much as possible, motion blur or no motion blur. De-blurring has no direct link to TSR except that variational methods can be used for de-blurring.

5.3.6 Algorithm for Frame Doubling

Our framework handles any reasonable frame rate conversion.[4] For testing we have chosen just to implement a frame rate doubler, but at the end of this chapter we will discuss how to implement algorithms for other conversion ratios, which turns out to be mainly a practical problem fairly easy solved. Here we look at the implementation of a frame doubler minimizing the energies in (5.5), (5.6) and (5.7) according to (5.3).

Let us start with the flow energy minimization. After exchanging the E_2-term of the flow energy in (5.6) with the E_2-term from (5.7), the flow Euler-Lagrange equation is derived. It is implemented numerically along the lines given by Brox *et al.* in [9] and by Lauze in [64] and minimized iteratively using a Gauss-Seidel solver with a fixed point approach to linearize the otherwise nonlinear system.

The same general solution is used for the intensity energy. We have that the optical flow constraint is approximated by the brightness constancy assumption, that is

$$\pounds_{\vec{V}} u = \nabla u \cdot \vec{v} + u_t = \vec{V}^T \nabla_3 u \approx u(\mathbf{x}, t) - u(\mathbf{x} + \vec{v}, t+1)$$

with the discretization suggested by $u(\mathbf{x}, t) - u(\mathbf{x} + \vec{v}, t+1)$. We define $A(u) = 2\psi'(|\nabla u|^2)$ and $B(u) = 2\psi'(|\pounds_{\vec{V}} u|^2)$. Then the gradient of the energy in (5.5) is

$$G(A(u), B(u)) := \frac{\partial E}{\partial u} = -\lambda_s \mathrm{div}_2 \left(A(u)\nabla u\right) - \lambda_t \mathrm{div}_3 \left(B(u)(\pounds_{\vec{V}} u)\vec{V}\right) \tag{5.8}$$

where div_2 and div_3 are the 2D and 3D divergence operators respectively. Discretization is performed as described for deinterlacing and video super resolution in Chapters 3 and 4. Equation (5.8) set equal to zero is the intensity Euler-Lagrange equation. It is unfortunately nonlinear with $A(u)$ and $B(u)$ being the nonlinear terms. In order to linearize the system so that we can apply a Gauss-Seidel solver to it, we use a fixed point approach. $A(u)$ and $B(u)$ are only updated in each of a number of outer fixed point iterations. For each outer iteration we run a number of inner iteration in which the values of $A(u)$ and $B(u)$ are now just constants (fixed) and the system thus linearized, enabling the use of a Gauss-Seidel relaxation solver.

In our multiresolution settings, on each level k of the pyramid, we first compute the forward and backward flows, $\vec{v}_0^f$ and $\vec{v}_0^b$, of the original input sequence u_0 (resized to the size of the current level), minimizing (5.6) (with or without the GCA included in E_2) in which the resized input sequence u_0 simply replaces u.[5] This is to have a highly reliable anchor flow when calculating the flows $\vec{v}^f$ and $\vec{v}^b$ of the full output sequence. At the given level of the pyramid, k, we then initialize intensities and the flows of the new frames by resizing the intensities and flows calculated at the above coarser level $k+1$. Then we calculate from these initializations the flows by minimizing the energy in (5.6), again with or without the GCA in E_2. Next we calculate u at level k by minimizing the energy (5.5) knowing $\vec{v}^f$ and $\vec{v}^b$ and using the resized intensities from level $k+1$ as initialization of u in the new frames, just as when calculating $\vec{v}^f$ and $\vec{v}^b$.

[4]We have already found frame doubling to be the worst case scenario, but doing e.g. 25 fps from 1 fps video or similar is what we would consider unreasonable, as good results would be unrealistic in case of complex motion in the scene.

[5]This corresponds to minimizing $E(\vec{v})$ over the domain K instead of minimizing it over Ω.

The resizing function (*resize*) we use to initialize flows and intensities at level k from the values calculated at the above level $k+1$ is a simple spatial down- and upscaling function for multiresolution schemes as given in [11]. We also use this *resize* function to downscale u_0 to the given level k to always have the best representation of the original frames at the given level.

The recalculation of the flow of the original frames, $\vec{v}_0^f$ and $\vec{v}_0^b$, is important, because it rids us of the assumption that the flow is constant/linear and can just be halved when we insert new frames. When the flows in the original frames is more precise, the flow we calculate in the new frames and actually use in the intensity calculations, also becomes more precise and reliable.

The use of a multiresolution schemes is considered essential when doing variational flow calculations. In TSR, calculating both flow and intensities at each level solves the hen-egg problem of what comes first in a new frame: The flow or the intensities. Thus we iteratively improve first one and then the other to get simultaneous computations and optimize our solution. By using a small scale factor between pyramid levels ensures a very good initial guesses of the intensities and flows in the new frames at each level. Using further integration to make the calculations 'more' simultaneous (as it is done for video super resolution in Chapter 4) by alternating between minimizing $E(\vec{v})$ and $E(u)$ internally on each level in the pyramid is unlikely to improve output quality when using small scale factors in the pyramid.

At the coarsest level at the top of the pyramid we do not have a $k+1$ level to initialize our data from and thus have to use temporal initialization (inferior to $k+1$ initialization). For the flow calculation we have chosen to do frame averaging of both flow and intensities. If the new frame is located at time n and the two know frame are at time $n \pm 1/2$ then

$$\vec{v}(\mathbf{x}, n) = \frac{\vec{v}_0(\mathbf{x}, n-1/2) + \vec{v}_0(\mathbf{x}, n+1/2)}{2}$$

and

$$u(\mathbf{x}, n) = \frac{u_0(\mathbf{x}, n-1/2) + u_0(\mathbf{x}, n+1/2)}{2}.$$

Since the spatial size of the frames at the top level is only a fraction of the size of the frames at bottom level, even very large motion will be downscaled to be very small at the top level, and thus the initialization will be a good approximation of the actual values. Even though the flow we compute at the top level is of subpixel (or close to) size, we still use it to re-initialize the intensities by simple interpolation along the flow

$$u(\mathbf{x}, n) = \frac{u_0(\mathbf{x}+\vec{v}^b, n-1/2) + u_0(\mathbf{x}+\vec{v}^f, n+1/2)}{2}$$

before we minimize $E(u)$. As the top level is only a very crude estimate of the finest level at the bottom and still has to go through many corrections down through the pyramid, this initialization should suffice.

The algorithms we use is (leaving out the special initialization case at the top level):

At each level from the top, coarse to fine, for k = *levels* until k = *1*:

1. Calculate the forward and backward flows, $\vec{v}_0^f$ and $\vec{v}_0^b$, of the resized original input sequence $u_{0,k}$ minimizing (5.6) with/without E_2 from (5.7).

2. Initialize new frames: $u(\mathbf{x}, t, k) = resize[u(\mathbf{x}, t, k-1)]$ in the domain D.
3. Initialize forward and backward flows of new frames:
   $v(\mathbf{x}, t, k) = resize[\vec{v}(\mathbf{x}, t, k-1)]$ in the domain D.
4. Calculate the flows $\vec{v}^f$ and $\vec{v}^b$ of the output sequence u minimizing (5.6) with/without E_2 from (5.7).
5. Calculate new frames in $u|_D$ by minimizing (5.5).

5.4 Experiments

In our tests we will focus on the major problem caused by having to low frame rates in image sequences: Unnatural motion. The problem is mostly caused by camera pans on scenes containing high contrast edges. By doubling the frame rate we will make the apparent motion of these high contrast edges smoother and (partially) reestablishing the phi-effect.

As discussed in the previous section we have implemented our frame doubling algorithm in such a way that we can test it in two versions: The first includes the gradient constancy assumption in the flow calculations and is expected to give the most correct results as both flow and intensities are subject to minimal blurring when the GCA sharpens the flow. The second method without GCA in the flow is expected to blur the flow more and following from that, also blur the intensities more. The version without GCA is, when implemented in a version where the GCA is not turned out by setting its weight to zero, faster. We test it in the hope that its results, although expected to be more smooth, will be convincing to the human viewer and not be perceived as unsharp (as discussed in section 5.2.2). If we get results with different sharpness in the new frames, it will help us say something about blur acceptance in frame doubling. Before we start to look at our frame doubling results, there are some topics we wish to discuss first.

5.4.1 Test Material

We have tested on a few homemade sequences as well as on cutout sequences of real world motions pictures on standard PAL resolution (720×576 pixels, 25 fps, telecined) DVDs. We only process the luminance channel, but the extension to full color processing is discussed in Section 2.4 of this thesis. The cutouts used are of areas with high contrast edges in motions and the sequences chosen were experienced to have jerky motion when watched in their full frame PAL versions on a 43" Pioneer plasma screen with a viewing distance of 2 to 2.5 meters. The frame rate of the screen is unknown, but the frame rate up-conversion from 25 fps is unlikely to be motion compensated due to the experienced jerky motion. The jerkiness in the test sequences was also confirmed by played back the 25 fps sequences on a LCD PC monitor with 75Hz refresh rate using frame repetition.

Online Video Files

All inputs and results given in the figures in Section 5.4.5 (Frame Doubling Results) are also given as video files (*.avi) online at: `http://image.diku.dk/`

		Without GCA	With GCA
Multiresolution	Scale factor	1.04	1.04
	Levels	55 or 75	55 or 75
Flow	Fixed point iterations	10	5
	Relaxation iterations	40	20
	λ_3 in (5.6)	30	100
	λ_2 in (5.7)	1	1
	γ in (5.7)	0	100
Intensities	Fixed point iterations	5	5
	Relaxation iterations	10	10
	λ_s in (5.5)	1	1
	λ_t in (5.5)	50	50

Table 5.1: Optimal parameter settings for variational TSR. The parameters are discussed in details in section 5.4.2. The eleventh parameter of the algorithms is the convergence threshold set to 10^{-7} in all tests.

`sunebio/TSR/TSR.zip` [58]. References to specific files are given when they are discussed in Section 5.4.5 and all files are named with the same name as the input test sequence and the method used on them. The shareware AVI video editor VirtualDub is also given in the online material.

5.4.2 Parameters

As with any advanced method there is a number of parameters that needs to be tuned in our algorithms to optimize performance. To be exact there are eleven parameters. In our case optimal performance is optimal output quality, we have left tuning for lower running times for later, but will discuss the issue in this section. Through extensive parameter testing we have optimized two sets of parameters for variational frame doubling TSR; with and without GCA in the flow. Our experience from doing deinterlacing and video super resolution (see Chapter 3 and Chapter 4 of this thesis) together with literature on inpainting and optical flow calculation ([9], [12], [14], [64] etc.) enabled us to give qualified guesses on parameters. From these initial guesses we could then try to push each parameters in both directions making our parameter tuning deterministic. Testing just three different settings of each parameter in all possible combinations would result in $3^{11} = 177,147$ different test results for evaluation.

The settings we decided on were optimal (given our obviously incomplete search) can be found in table 5.1 for both versions of our algorithm. We see that settings for the intensity energy minimization is the same for both algorithm versions, we did of course test other settings, but the given settings optimizes output quality. The weight ratio temporal to spatial diffusion is high in favor of temporal diffusion which ensures that spatial diffusion is only used when temporal information is highly unreliable. Lowering the ratio from 50:1 to 20:1 gave similar results when evaluating on video, but judging from the stills, there where minor degradations, thus we recommend λ_t:λ_s = 50:1. It is possible that fewer intensity iterations will give results similar to those we get now, but as we consider lowering the number of iterations to be pure running time optimization, we have (not yet) tested this tuning option thoroughly.

The number of flow iterations needed are higher than the number of intensity iterations needed, which indicates a larger complexity in the flow calculations. We did conduct tests with fewer flow iterations, but we are close to the bottom limit as is. Experiments of running more iterations than given in Table 5.1 gave no visual improvements – with one exception: While tuning we found that doing 20 fixed iterations instead of 10 in the flow calculations without GCA gave very small visible improvement, but only when results where viewed as still, thus we are confident that 10 iterations are enough here. Even though we appear to be at convergence with the given number of iterations, we never reached the convergence threshold of 10^{-7}, which would break out of the loops and stop iterations. Thus a higher value of convergence threshold might be of more use, if this parameter is to live up to its name.

Table 5.1 also shows that without GCA in the flow we need more iterations to get optimal flows. This is because fewer point give reliable flow information when only evaluating brightness constancy, which increases the need for flow diffusion by the regularization term on the flow (E_3) – and this takes extra time. The E_3-term is weighed 30 times over the OFC when we do not use the GCA, but is given the same weight as the GCA when the GCA is used (with the OFC nearly neglected due to the λ_2:γ = 1:100 ratio). Since we have implemented flow calculations without GCA by setting $\gamma = 0$, we cannot say if the version without GCA in the flow is faster than the version with GCA in spite of its need for four times as many iterations, but we do expect it to be faster. It would under all circumstance be less complex to implement e.g. in hardware.

The number of levels in the multiresolution pyramid is set to either 55 or 75 depending on the frame size of the given image sequence with a 1.04 (coarse to fine) scale factor between the levels. The low scaling factor ensures good information transfer down through the pyramid, but we have not tried increasing to values above 1.1 (and only tried the value 1.1 on one sequence) but doing so could optimize running times as each level would shrink in size. Whether larger scale factor will decreasing the output quality is unknown. The maximum flow magnitude (relative to the frame size) should be scaled to (close to) subpixel at the top level. With larger scale factors the number of levels in the pyramid could also be decreased, but this would only give a minor speedup as we would remove only the very small coarse levels at the top of the pyramid. It is more likely that we would get a speedup from the lowering the number of iterations at all levels but the finest bottom levels of the pyramid (and possibly the top levels to cope with the inferior initializations used) but again this is a speedup optimization and is left for later.

We also conducted experiments using zero as initial values for both flows and intensities in the new frames at the top level. Given the many levels we use, the error introduced was corrected down through pyramid, showing great robust against bad initializations. The robustness of our algorithm is also enforced by the fact that we compute the flows on the original frame only on each level.

5.4.3 Other Methods in Test

To get a perspective on the quality of our algorithms (with/without GCA in the flow) we will also give results from doing frame averaging. We would also have liked to compare our method with other advanced motion compensated methods to get a really strong benchmark, but this has not been possible as discussed in Section 5.2.8 – Benchmark in Testing.[6]

5.4.4 Evaluation Strategy

As the human visual system is the final judge when evaluating the enhancement achieved by upscaling, here TSR, we will focus on subjective results. To give a broad validation, we have also given objective results. The problem of finding an objective measure that really qualifies as a reliable measure of the quality as perceived by the HVS has been discussed in Section 2.3 of this thesis.

One cannot evaluate the quality of frame doubling or TSR by looking at stills alone. Stills are valuable as they give detailed view of the quality of results, the degree of smoothing and other artifacts. It is imperative to evaluate on video as well to get the true realtime experience of the results. It might be that the some blur in the new frames is not seen in realtime playback and that sharp artifacts, which are often sensed much clearer by the eye than blurring, also disappear. Video evaluation is also essential to judge if motion portrayal has become natural.

In our tests we have doubled frame rates from 25 to 50 fps, which should enable viewing the video examples in the online material [58] at any modern PC screen at refresh rates at 50 Hz or above. For comparison we have also given the 25 fps input sequences in [58]. Playing the videos in Windows Media Player clearly illustrates the difference in qualities that are described in frame doubling results section (Section 5.4.5). As the example sequences given are rather short, looped playback can be used. This is of course not the true way of seing the results – in looping one might spot something not seen in just one pass – but it can help do detailed analysis. Detailed analysis can also be done by frame by frame inspection of the result for which we recommend using VirtualDub given in [58].

As objective measures we have used the mean square error (MSE) and the peak signal to noise ratio (PSNR). Using the notations given in Section 5.3, the MSE is

$$MSE = \frac{1}{N}\sum_{\Omega}(u - u_{gt})^2 \tag{5.9}$$

where u is the frame doubled output and u_{gt} is the ground truth. We sum over all pixels of the sequence (also frames from the input kept unchanged in the output). The PSNR is

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \tag{5.10}$$

with inverse ordering of the MSE. PSNR is measured relative to the maximum possible grey value, 255.

5.4.5 Frame Doubling Results

In Figure 5.2 results for the sequence `Square` is given. `Square` has 50×50 frame size, is 5 frames long in the input and 9 frames in the output. The 10×10

[6]If we get the chance to continue work on TSR, doing a thorough TSR benchmark would be high on the list of priorities.

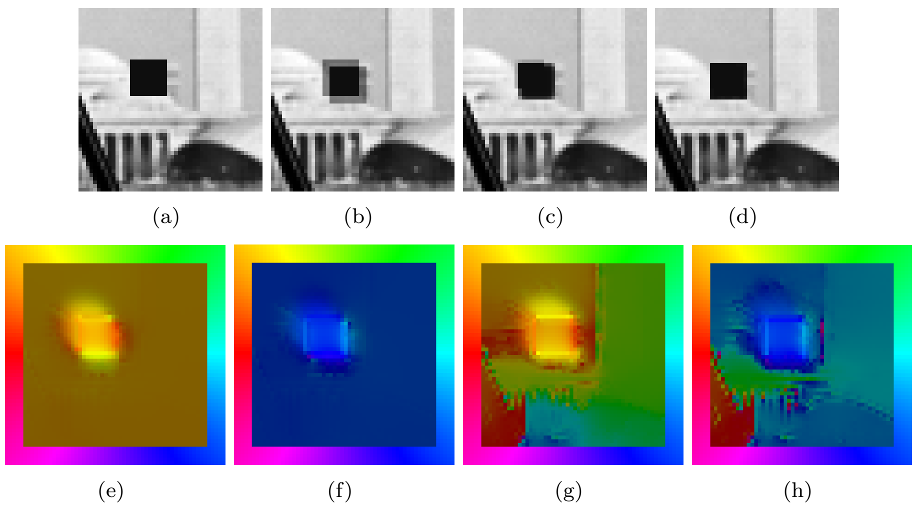


Figure 5.2: Frame Doubling on the 50×50 sequence `Square`. (a) Original frame 3, now frame 5 of the double frame rate output. The new frame 6 of the out put created by (b) frame averaging, (c) variational TSR without GCA, and (d) variational TSR with GCA. Optical flows computed in variational TSR: (e) backward from frame 6 to 5 without GCA, (f) forward from frame 6 to 7 without GCA, (g) backward from frame 6 to 5 with GCA, and (h) forward from frame 6 to 7 with GCA. In this color representation, the hue value gives the flow direction as coded at the frame boundary and the intensity (normalized in [0.5-1]) gives the flow magnitude.

square moves 2 pixels/frame diagonally down to the right in the output. This sequence will test the ability to handle local object motion and (dis)occlusions.

Frame averaging as shown in Figure 5.2(b) clearly creates a double, semi-transparent square. Variational TSR without GCA does not perfectly recreate the square as seen in Figure 5.2(c), but when the result is truthfully watched as video, the square is not perceived as unsharp in any way and the motion looks fluent compared to the jerky and unnatural motion seen in the input. Played as video there is no way to tell apart the two variational TSR results apart (with/without GCA) even though there is a distinct difference in the still outputs as variational TSR with GCA perfectly recreates the square as seen in Figure 5.2(d). The frame averaging output is on the other hand clearly jerky and has a trail of the square. The outputs are given as the videos `SquareFrameAv.avi`, `SquareVarTSRnoGCA.avi` and `SquareVarTSRwithGCA.avi`, all are at 25 fps and should be watched zoomed to 200% or sitting very close to the screen as they are very small.

The flows computed in the variational TSR algorithm are shown in Figures 5.2(e)-(h). The background in `Square` has zero motion and both variational TSR versions gives flows that are approximately zero as expected, but the flows computed with GCA has large directional variation due to the strong direction coding in our color visualization of the flows. Since the flow are ≈ 0 this has no influence on the intensity results. A (dis)occlusion trail can be seen in the flows, which in the non-GCA version gives some artifacts as shown in Figure 5.2(c).

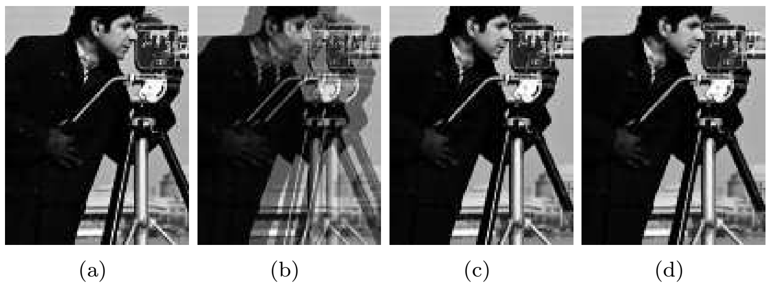


Figure 5.3: Frame doubling of the 130 × 100 sequence `Cameraman Pan`. (a) original frame 5, now frame 9 of the double frame rate output. New frame 10 by (b) frame averaging, (c) variational TSR without GCA, and (d) variational TSR with GCA.

Under very careful still inspection of the GCA version, the (dis)occlusion causes a slight smoothing of the background in the trail where the non-GCA version has artifacts. The smoothing is due to the use of spatial diffusion where the flow is not highly reliable. We also see an nice filling in of the flow (by the E_3-term) in the center of the completely flat square, and even though it is very hard to detect visually in the flows, the GCA version is closer to the correct flow magnitude and direction at the corners of the square causing better intensity result as seen in Figure 5.2(d).

The sequence `Cameraman Pan` is a pan (10 pixel/frame in the input) across the image `Cameraman` in a 130 × 100 window. Results of frame doubling (25 to 50 fps) is given in Figure 5.2. Videos (with descriptive names) for the input and all three outputs can be found in the online material [58].

On the sequence `Square` frame averaging was performing bad but not unacceptable. On `Cameraman Pan` the performance of frame averaging is unacceptably bad as it is should become clear from looking at Figure 5.3(b). Artifacts this bad is not hidden when viewing the result as video and the motion seems if possible even more jerky than the motion in the 25 fps input. The motion of the two variational TSR outputs are much smoother and appears very natural. In Figures 5.3(c) and 5.3(d) it is also seen how the new frames produces with variational TSR are very similar to the original in Figure 5.3(a) except for a slight smoothing seen only in the stills. The good performances of our motion compensated methods are only natural as the motion is a uniform global pan that any optical flow method should easily estimate correctly.

Some minor (dis)occlusion errors occur at the right and left frame boundaries of `Cameraman Pan` when details leave or enter the frame, e.g. the shadow extension of the upper camera lens in Figures 5.3(c) and 5.3(d). These errors are only spotted during video playback if one looks for them or happens to focus on that particular part of the frame. The problem does not lie in the flow calculation, the flow is estimated correctly all over the frames, but is caused by the fact that, if one of the flows in a pixel (backward or forward) points out of the frame, then we use Neumann boundary conditions to find a replacement value of the pixel sought for in neighboring frame. This creates a temporal gradient of high magnitude if the next frame pixel found is very different from the pixel currently being processed (which is the case where detailed regions are moving)

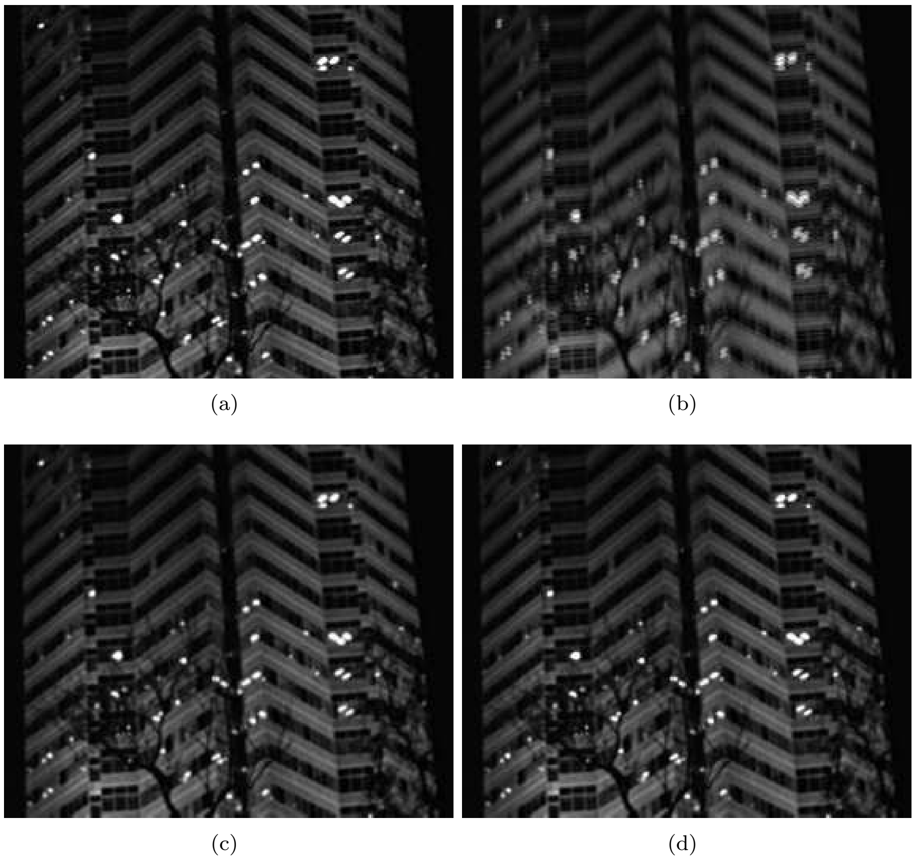


Figure 5.4: Frame doubling on the 284 × 236 sequence `Building`. (a) original frame 1 and still frame 1 of the double frame rate output. New frame 2 by (b) frame averaging, (c) variational TSR without GCA, and (d) variational TSR with GCA.

and leads to decreased temporal diffusion and increased spatial diffusion – as can be deduced from Equation (5.8). The implicit temporal edge adaptiveness of total variation thus fails (due to boundary conditions and choices made in the implementation) and to fix the problem one could (and should) hard code a boundary flow reliability measure simply shutting of temporal input from any of the two flows pointing out of the frame. To be fair to our framework and its total variation based modelling, the use of forward and backward flow is introduced as an assumption of bidirectional flows in the numerical scheme and is not modelled directly in the framework (the $E(u)$ part).

The sequence `Building` is a 284×236 cutout of a PAL DVD (telecined from film). The scene chosen has a camera tilt down a building, making the motion close to global but with discrepancies from uniform global translational motion due to the depth of the scene and the fixed camera viewpoint. The motion of the camera is also not constant, it varies slightly up and down in speed throughout the sequence. The structure of the building and the contrast obtained in the recording makes the apparent motion of the 25 fps input jerky. Results on frame doubling `Building` can be seen in Figure 5.4 and in the online videos [58].

Frame averaging on `Building` blurs the new frames quite a lot as can be seen in Figure 5.4(b). During video playback this blurring is not sensed, but the motion is still as jerky as in the input and the lamps seen inside the building seems to flicker as they are blurred to middle grey in the new frames. The human visual system is very sensitive to flickering as discussed earlier in this chapter. The flicker is perceived as a temporal change and temporal change sensitivity is larger away from the fovea. Since the lamps are spread out over the frame, some will be projected to the off-fovea part of the retina when viewed at a large viewing angle (large screens, short viewing distances). The contrast between the lamps and their surroundings is large and together with the flicker and the jerky motion mentioned above, the frame averaging result is very annoying to watch.

The motion portrayal in the two variational TSR results is natural and no flickering occurs. As seen in Figures 5.4(c) and 5.4(d) the new frames are a bit smoothed compared to the original frames, but this is not noticeable in the videos.

From the same movie as we got `Building` from, we have taken the sequence `Control Panel` (256×220 cutout) with a fast camera pan and complex motion (the person walking behind the control panel and being tracked by the camera). Results are given in Figure 5.5 and in the videos [58].

As with `Building` the frame averaging result on `Control Panel` still has jerky motion and flickering seen when playing the video (found in the online material [58]) and as Figure 5.5(b) shows the new frame averaged frames are also blurry.

For the panning motion on the control panel, Figures 5.5(c) and 5.5(d) show that we still get new frames only a fraction smoother than the original frames when applying variational TSR, and as can be seen in the accompanying videos the apparent motion becomes natural and is no longer jerky as in the input. In the top third of the frames of `Control Panel` we have the complex motion of the walking person. The right arm with creases and sewings in the shirt is correctly reproduced in the new frames although with some denoising/smooting on the sleave. But the west worn by the person and swinging back and forth as he walks is not reproduced correctly as can be seen clearly in new frames 6 and 10 in both of the variational TSR videos. (Try to go back and forth between frame 5-7 and 9-11 manually in the program VirtualDub given in the online material [58].) The changes in the motion of the west (the motion acceleration) and the complexity of the motion with many and fast (dis)occlusion is too much for our variational optical flow schemes in their current versions. Analysis of the problem and suggestions on how to solve it will be discussed in section 5.5.1.

The sequence `Boat` (320×306 cutout) taken from a PAL DVD (film origin) has an even faster pan than `Control Panel` and object motion as well. The motion is very stuttering when the 25 fps input sequence is played back (`Boat25fps.avi`) and `Boat` has the most unnatural motion of all the sequences we have run tests on. Results are given in Figure 5.6 and accompanying videos.

Again the frame doubling result is of poor quality as seen in Figure 5.6(b) and the video (`BoatFrameAv50fps.avi`), and the two variational TSR schemes produce high quality results, only slightly smoothed (Figures 5.6(c) and 5.6(d)) but with natural motion portrayal as seen in the videos.

Repeated watching of the variational TSR results on `Boat` gives a sense of

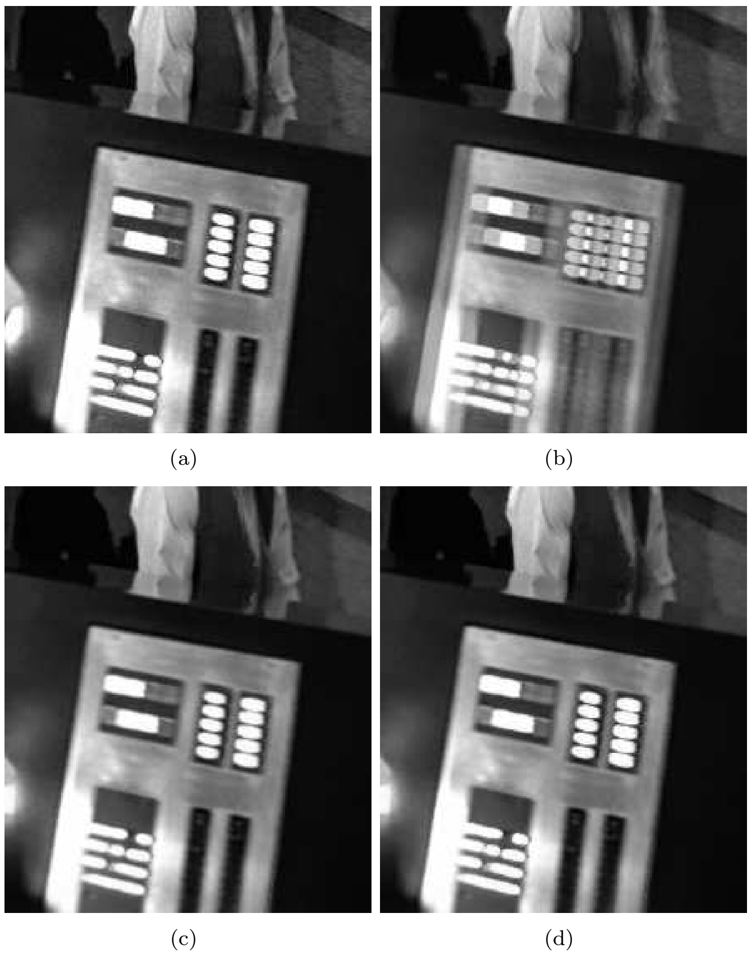
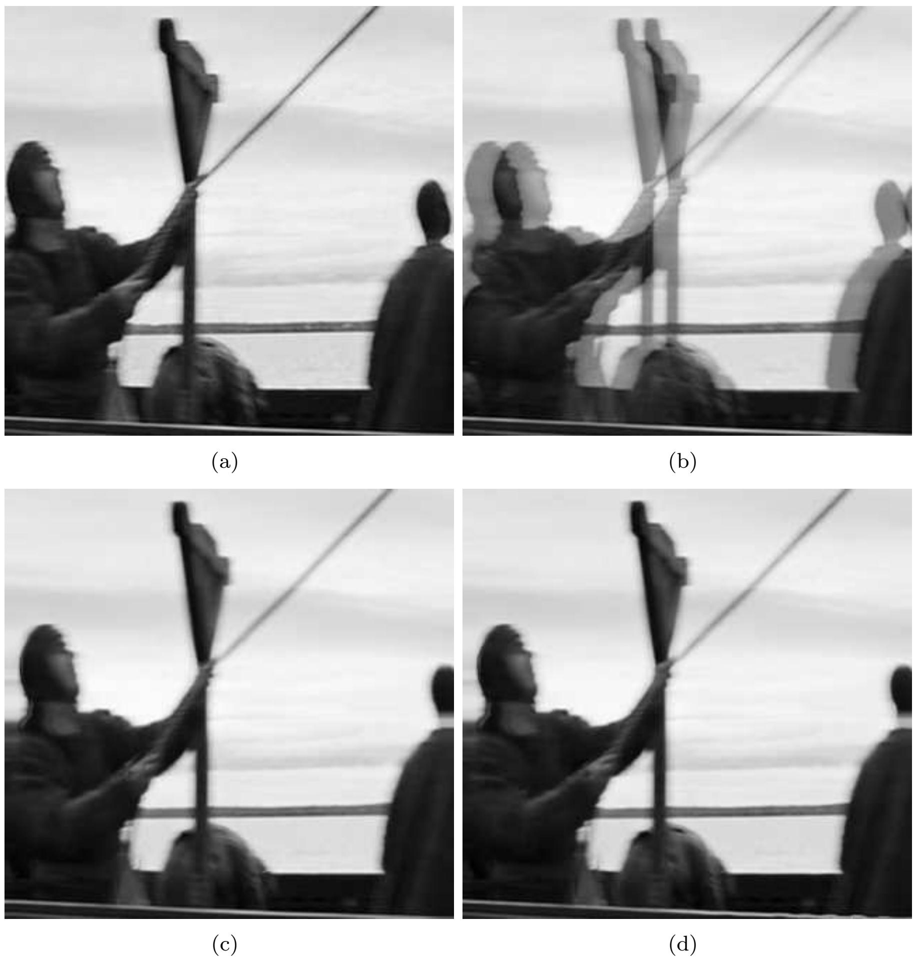


Figure 5.5: Frame doubling on the 256 × 220 sequence `Control Panel`. (a) original frame 4, now frame 7 of the double frame rate output. New frame 8 by (b) frame averaging, (c) variational TSR without GCA, and (d) variational TSR with GCA.

a slight stutter in the motion, indicating that even though the edges in the sequence are a bit unsharp due to motion blur in the input, the contrast is so high that viewing it on really large and bright screens might require 75 fps to get a smooth and fully natural motion portrayal. For frame tripling we will need to change our current frame algorithm only suited for frame doubling. The design of a generalized variational TSR scheme producing arbitrary frame rates is discussed in section 5.5.2.

5.4.6 Objective Evaluation of Frame Doubling Results

We have computed objective results for four of the test sequences evaluated subjectively in the previous section. For the two artificially created sequences `Square` and `Cameraman Pan` we have created the ground truth frames corre-



(a) (b) (c) (d)

Figure 5.6: Frame Doubling on the 320×306 sequence `Boat`. (a) Original frame 2, now frame 3 of the double frame rate output, new frame 4 by (b) frame averaging, (c) variational TSR without GCA, and (d) variational TSR with GCA.

sponding to the new frames in the frame doubled outputs. For the real sequences `Building` and `Control Panel` we have taken out every other frame of the inputs and used these shortened sequences to compute frame doubled outputs the length of the original sequences now acting as ground truth. This means larger panning/tilting motions from frame to frame as we now do 12.5 to 25 fps frame doubling. Since frame repetition is not really worth comparing with other results in stills, and since it is what you get from playing the 25 fps input on a monitor capable of playing 50 fps or more, we left it out of the subjective evaluation but have included it here.

As the results in Table 5.2 show, our variational TSR algorithms outperforms frame averaging as it was also the case in the subjective evaluation. It is also no surprise that in the presence of motion, frame repetition is clearly the worst performing frame doubling algorithm in terms of objective results.

		`Square`	`Cameraman Pan`	`Building`	`Control Panel`
Frame repetition	MSE	158.4	1887.9	462.5	1369.8
	PSNR	*26.13*	*15.37*	*21.48*	*16.76*
Frame averaging	MSE	82.54	1208.3	287.8	849.9
	PSNR	*28.96*	*17.31*	*23.54*	*18.84*
TSR without GCA	MSE	13.05	**39.47**	**13.82**	**76.59**
	PSNR	*36.97*	***32.17***	***36.73***	***29.29***
TSR with GCA	MSE	**8.97**	107.9	16.44	96.87
	PSNR	***38.60***	*27.80*	*35.97*	*28.27*

Table 5.2: Objective evaluation of the four frame doubling methods in test: Frame repetition, frame averaging and variational TSR without/with GCA. MSE (5.9) and PSNR (5.10) scores are given for the four test sequences `Square`, `Cameraman Pan`, `Building` and `Control Panel`.

Whether it is worse than frame averaging is however a question up for debate. Subjectively frame repetition does not change the jerky motion present in the input while frame averaging slightly smooths it. But frame repetition does not introduce any flickering, double exposure effect or smoothing in the new frames as frame averaging does. The smoothing might give less objective error, but frame averaging is worse than frame repetition on our data set as it does too little to remove the jerky motion and introduce visually annoying artifacts.

Returning to the far better variational TSR frame doublers, their objective measures are very close to each other as it was also the case when we evaluated their performances subjectively. The use of the GCA help in the case of object motion. Variational TSR with GCA gives the best result on the sequence `Square` which corresponds well with the subjective results given in Figure 5.2. For the two sequences `Cameraman Pan` and `Building` dominated by global motions, the non-GCA version is objectively slightly better than the GCA version, which might tend to overfit the flows. On the sequence `Control Panel` the non-GCA version produces a smoother flow field and thus the intensity output is also somewhat smoother, which helps dampen the problems with wrong flow estimations on the walking person as mentioned in the subjective evaluation of `Control Panel`.

In a subjective evaluation the ground truth sequences we use to get objective measures might not score better than some corresponding sequence produced using a good frame doubling algorithm. The problem with many objective measures including MSE and PSNR is that they give a global average and does not give special weight to local problems which might be judged very annoying by the human visual system.

From our combined tests we can conclude that variational TSR without GCA performs slightly better or the same as TSR with GCA in cases where the sequences are dominated by global flow (camera motion). From the results on the sequence `Square` it is indicated that TSR with GCA is better in case of object motion, but there are more fundamental problems with the flow than the minor differences in performance with/without GCA. These problems were observed in the sequence `Control Panel` and will be discussed in the next section, and from the improvements suggested we believe that TSR with GCA will give the best results on sequences with complex motion.

It is clear that our motion compensated variational TSR frame doublers are producing outputs far superior to the outputs from the simple methods frame averaging and frame repetition, but we need to benchmark variational TSR against other motion compensated TSR algorithms to show its full potential. The problems of doing such a benchmark was discussed in section 5.2.8.

5.5 Discussion

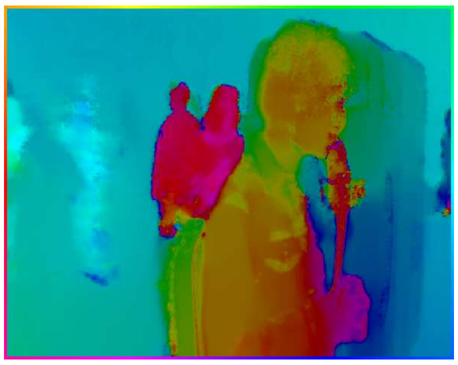
5.5.1 Improving Variational Optical Flow in Motion Compensated Upscaling

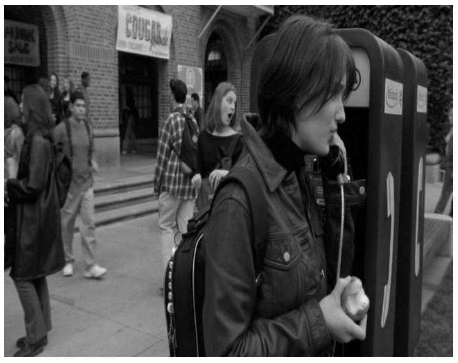
Scenes with complex motion can cause problems even to advanced optical flow algorithms. We will look at some examples of variational optical flow computed on sequences with complex motion. The flows have been computed by minimizing the flow energy with gradient constancy assumption as given in Equations (5.6) and (5.7).

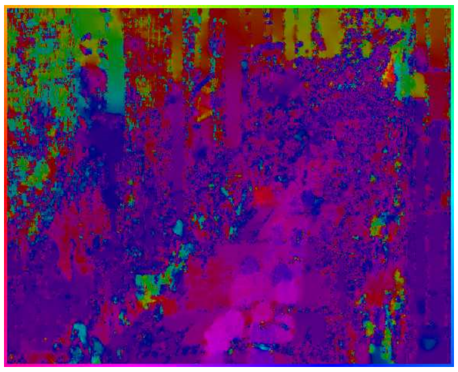
First, in cases where there is many different motions in a scene, e.g. crowd scenes, a highly segmented optical flow needs to be computed. In Figure 5.7(b) a relative complex scene, `School Yard`, with multiple motions including a camera pan is shown and in Figure 5.7(a) we see how the variational optical flow is nicely segmented – the two persons in the middle (red in the flow illustration) do actually have identical projected motion. In the sequence `Street` also used in Chapter 4 the motion is even more complex and many of the object in motion (far away cars and pedestrians) are too small to be segmented in the flow, see Figures 5.7(c) and 5.7(d). As shown with the flows of the sequence `Square` (see Figure 5.2) the flow computations with GCA will produce very fragmented flows even in detailed regions in with no or very little uniform motion, e.g. the stationary background in `Square` and the close to stationary buildings part of `Street`. To get smoother (and better looking) flows one could do some Gaussian pre-smoothing of the image sequence, but it might cause a loss of details in the flows.

Looking only at the objects with significant flow magnitudes in `Street`, the cars in the front, the van turning right around the corner in the middle of the frame, the Canadian flag to the upper right and the US flags on the left, they are all well segmented with reliable flows. For both examples in Figure 5.7 we have been able to compute artifact-free video super resolution results.

It is important to have a robust motion compensated algorithm that switches off temporal input when flows are unreliable, but the limitations of the human visual system will also help: In this kind of complex scenes the HVS will not be able to track all the motions and thus we might 'get away' with producing suboptimal outputs. Still optimal results require precise and reliable flows. Even if there is only one or a few motions in the sequence, but one of them is of a type that is not included in our motion modelling and handled in a way that artifacts will appear, the HVS will most likely spot the problems. The non-modelled motion could be transparent motion, whirls, some cases of accelerated motion, and specific to block matching algorithms; anything not translational


(a)


(b)


(c)

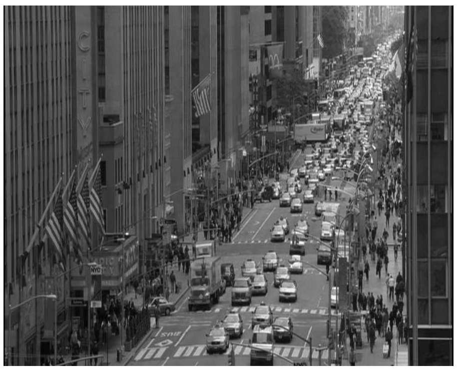

(d)

Figure 5.7: Variational optical flow with gradient constancy assumption. (a) forward flow of frame 13 in the sequence `School Yard` shown in (b). (c) forward flow of frame 2 in the sequence `Street` shown in (d). Both sequences are from movies telecined anamorphically to (widescreen) PAL DVDs.

at block size resolution.

A problem not handled very well in our variational flow algorithms both with and without GCA, is fast dynamic changes; flows with large accelerations. An example is the sequence `Keyboard` where the fast up-and-down motion of the typing hands causes problems. Four of the twenty frames in `Keyboard` are shown in Figure 5.8 and when we compute the flow (still with GCA as in the previous examples) of the full sequence as one volume, we get a flow that changes very little over the duration of the sequence, the true optical flow of the hands is not found. This is most likely due to the fact that we use local 3D spatiotemporal regularization on the flow, $\int_\Omega \big(\psi(|\nabla v_1|^2) + \psi(|\nabla v_2|^2)\big)dx$. This regularization term will fill in information from local temporal neighbors and in theory switch off the temporal diffusion if there are edges in time (the moving object has moved away). But for several sequences we have tested on, the flows stay the same over time in a given spatial location even though the object causing the motion is only present in that location in part of the sequence. `Keyboard` is one of these sequences, but `School Yard` where the correct motion fields nicely follow the objects over time, is not. We have used the 3D regularization on the

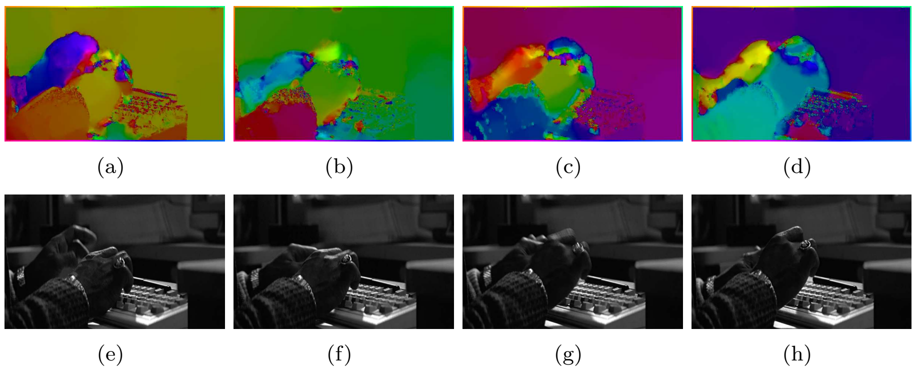


Figure 5.8: Optical flows with GCA on the 20 frame sequence `Keyboard` processed in 4 bites of 5 frame each. (a)-(d) forward flows of the second frame of each bite corresponding to frames 2, 7, 12 and 17 of the full sequence, the frames shown in (e)-(h). The flows are a somewhat oversegmented as the weight of the flow prior/regularization was set very low in this experiment ($\lambda_3 = 30$).

flow as it was reported to give better result than pure spatial 2D regularization by Brox *et al.* in [9] and by Bruhn *et al.* in [12].

To solve the problem of handling accelerated motion one could process the sequence in shorter bites. We tried to do so with `Keyboard`, we cut it into four bites of five frames each and the resulting flows are show in Figure 5.8. As the figure shows we now have dynamically changing flows and the problem is solved. Internally in each bite the flow is still close to constant over time, but this can be handled by making shorter bites, maybe even processing frame in pairs.

Solving the acceleration problem by bite wise flow processing in very short bites will give a loss of temporal long distance information propagation and time consistency achieved by processing longer bites iteratively. Using a sliding windowing function in time processing in overlapping bites might prevent some of this loss, but to get a sound solution to the problem, the solution should be formulated theoretically and from there implemented.

The 3D regularization (prior) we use, will in a multiresolution setting give an advantage with temporally slowly changing motions like zooms, exemplified by results on the `Yosemite` sequence given in [9] by Brox *et al.* where the variational optical flow algorithm using 3D regularization has lower angular errors than when using 2D regularization (and also outperforms any other algorithms tested on `Yosemite`). We have in our work on motion adaptive deinterlacing in [55] and [57] tested a 3D total variation regularization on intensities, but due to a lack of shutdown of temporal diffusion across temporal edges in case of motion, we abandoned it and used the split 2D and 1D prior instead as reported in Chapter 3 of this thesis. The assumption that a spatiotemporal volume is 3D seems to hold only in limited cases and image sequence volumes should be defined as 2+1D, separating space and time and using optical flow as the link holding them together (the '+'). Taking variational optical flow beyond `Yosemite` and other artificially generated test sequences and using it in general applicable motion compensated algorithms (e.g. temporal super resolution and other upscalings) we believe using 2D regularization will help solve the accelerations problem encountered with when using 3D regularization and we intend to test that in the future. We would also like to do the in-between of 3D and 2D priors, a local 2D+1D prior with a stronger temporal shutdown than the 3D prior or a 2D+1D prior including a $\pounds_{\vec{V}}\vec{V}$ derivative along the flow. The latter might – depending on its exact formulation – actually prevent motion acceleration, in that case some specific acceleration prior should be employed instead. Any improvement of our model keeping us from using bite wise flow processing to handle accelerations is welcome as it will improve temporal consistency.

5.5.2 Pointers to Generic Variational Frame Rate Conversion

Doubling the frame rate or doing any other integer multiplication of the frame rate is relatively straightforward, but any odd conversion like 25 to 24 fps, 25 to 72 fps or 24 to 60 fps is more requiring. Our framework allows arbitrary frame rate conversions, but instead of keeping old frame as every other new frame, the data term will have to specify a mapping from input temporal resolution to output temporal resolution, similar to the spatial super resolution constraint used in Chapter 4 of this thesis. Since the temporal point spread function would have to model the aperture time of the camera, one would always need to now the shutter speed used by the photographer for every recording. It would be more practical to let the data term map the nearest original frame in each direction to know where to find reliable data when creating the new frames. Lets us leave the more theoretical discussion and look at the implementation issues involved in doing generic (in frame rates) temporal super resolution.

To get TSR algorithm with arbitrary frame rate conversion, first the initializations need to be changed, which should be rather simple to do. Multiresolution schemes are a great help here, since we can always build a pyramid that downscales the image to a size where $|\vec{v}| \approx 0$ or at least $|\vec{v}| < 1$ and thus get a good initialization of our basic data.

Secondly we would have to change the nice, regular spaced temporal grid with grid size $\Delta t = 1$, to a more irregular grid, which mainly affects the calculations of temporal derivatives in the numerical implementation of the energy minimizations.

Thirdly, in the frame doubler we are in the fortunate situation to have two original frames close by as nearest neighbors to each new frame. When doing arbitrary frame rate TSR we might have more than one new frame between each pair of original frames and the question is to what degree should we ignore data in a neighboring new frame to get more reliable information from a further away original frame. Iteratively solving our problem, data in new frames become more and more reliable and neighboring new frames represents reliable information transported there from known input frames. With a good initialization from the level above in the multiresolution pyramid when using a small scale factor between levels, we can also gradually improve quality as we go down through the pyramid.

Finally one could consider if any original frame close to the position ($\pm\Delta t$) of a new frame should be used directly as that new frame. The maximum Δt_{max} we can allow depends on the maximum projected displacement Δx_{max} on the retina, which does not annoy the HVS by creating jumpy/jerky motion. Δt_{max} depends on Δx_{max} thought the maximum projected velocity Δv_{max} in the depicted scene: The larger the velocity, the smaller Δt_{max}. Δx_{max} is however not fixed, but depends on many factors such as contrast, brightness, viewing angle and so on.

5.6 Conclusion

In this chapter we have discussed the requirements put on the design of temporal super resolution algorithms by the human visual system, we have presented a novel idea of simultaneous flow and intensity calculation in new frames of an image sequence, and we have introduced a novel variational temporal super resolution method and implemented and tested a variational frame rate doubler in two versions.

Even though we do not always create perfect new frames, variational motion compensated temporal super resolution does provide high quality 50 fps video from 25 fps video without noticeable artifacts during video playback, thus reestablishing the pi-effect for the problem case of high contrast edges in motion. The framework presented also has the potential to be used for other frame rate conversion than frame rate doubling.

Chapter 6

Detecting Interlaced or Progressive Source of Video

This chapter is a collaboration with Kim Steenstrup Pedersen and my co-supervisor François Lauze and was originally published as [56]. Minor corrections have been done from [56] to improve readability.

In this chapter we introduce an algorithm – commonly known as a film mode detector – for separating progressive source video from interlaced source video. Due to interlacing artifacts in the presence of motion, a difference in isophote curvature can be measured and a threshold for effective classification can be set. This can be used in a video converter to ensure high quality output. We study two approaches.

6.1 Introduction

Many elements are needed to make a full video converter. Some of the most important elements are a deinterlacer, a spatial resolution up-converter (video super resolution) and a frame rate converter (temporal super resolution). The input video can be either interlaced or progressive [82]. In an interlaced video signal (broadcast or stored on e.g. DVD discs) one can have progressive video embedded, e.g. when the signal is of film source telecined to interlaced [82]. By doing a pull-down – that is recreating the original progressive frames from the interlaced fields – before further processing, interlacing artifacts can be avoided in progressive material as a deinterlacing would not necessarily remove all interlacing artifacts [55], [4]. The quality of interlaced material will in the presence of motion also suffer from just being merged to frames instead of being properly deinterlaced.

Thus determining the scan format of the input is vital for the further processing and the output quality. Hence another key element in a video converter is the input scan format detector. This element is often called film mode detection as film was earlier the only source of progressive material, but today progressive material can also originate from video and television cameras.

If the input source is DVD, the MPEG-2-codec facilitates flagging of video as either interlaced or progressive, which could make source detection obsolete.

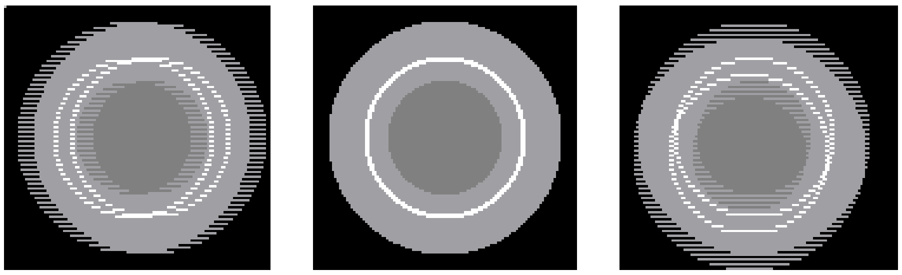


Figure 6.1: Interlacing artifacts: Serration, none (progressive) and line crawl

Unfortunately, it is far from sure that the flagging has been done correctly [76] and if the source is standard broadcast there is no flagging.

Some material like documentaries mixing video and film material and 'behind the camera' shows on movies mix progressive and interlaced material... Therefor it is important to have a film mode detection, that can switch from one format to the other relatively fast.

6.2 Theory

6.2.1 The Difference Between Interlaced and Progressive

To develop an effective algorithm for separating progressive source video from interlaced source video we need to establish exactly what the difference between the two formats is and how to measure this difference. The key to this lies in the motion of the image sequence.

Ideally one can just merge two consecutive interlaced fields to a frame, but this only works when there is no motion in the sequence. When motion is present it will give rise to the two types of artifact shown in Figure 6.1 and explained in [55] and Chapter 3 of this thesis. These artifacts are exactly what gave rise to the idea of the algorithm presented in this chapter.

Three topics have to be considered to get to the final algorithm and they are given in the following three sections.

6.2.2 The Measurement – Isophote Curvature

As can be seen directly from Figure 6.1, a lot of crenellation and serration appears in the merging of two interlaced fields in the presence of motion, whereas no artifacts arise from when merging two fields to their original progressive frame. We therefore suggest that isophote curvature of the image is a good measure of the difference between interlaced and progressive source video, as interlaced video will on average have a higher curvature. The equation for the curvature, κ, using image derivatives is

$$\kappa = \frac{I_x^2 I_{yy} + I_y^2 I_{xx} - 2 I_x I_y I_{xy}}{(I_x^2 + I_y^2)^{3/2}} \tag{6.1}$$

The image derivatives are computed using scale-space derivatives [61].

6.2.3 Measuring the Statistical Difference

To measure the difference between the curvature of interlaced and progressive video we build histograms of the curvature for sequences of a certain number of frames. To measure the actual difference, we use the Kullback-Leibler Divergence [34]

$$D_{KL}(P(\kappa), Q(\kappa)) = \sum_{\kappa} P(\kappa)(\log P(\kappa) - \log Q(\kappa))$$

as it puts weight on differences in the tail of a distribution. In our case that is where the high curvatures are represented and as can be seen in Figure 6.1, where we expect the major difference in curvature between interlaced and progressive source video. The histogram bins cover $|\kappa| \in [0, 100]$. To avoid 0-bins we use the Laplace-estimator of the probabilities and initialize all of the 101 equally sized bins with one sample each [25]. All $|\kappa| >= 100$ goes in the top bin.

6.2.4 Edges

From Figure 6.1 we see that the most significant information about the difference between interlaced and progressive can be found at edges in the frames. 5-10% of all pixels are on average detected as edges using a standard Canny edge detector, so if the edge detector takes less than 90-95% of the time a full frame curvature calculation takes, it lowers the computational cost of the algorithm.

6.2.5 Two Approaches to a Solution

The use of Kullback-Leibler Divergence (D_{KL}) as our measure implies the first idea for our algorithm, namely to build a distribution of curvatures from a lot of progressive material and then compare unknowns to it, thus we do classic classification with comparison to a learned distribution. We denote the known distribution of 'all' progressive material P. To measure the divergence from P using D_{KL} we take smaller bites of an interlaced stream of video and build a distribution and denote it Q. We also make distributions from bites of progressive video embedded in an interlaced stream and this is denoted Q'. For testing purposes 'the unknown' is of course known and thus also how to distinguish between Q and Q'.

To get directly comparable results, Q and Q' are generated in pairs from a progressive original. Interlaced is made by artificially removing every second line from the original and progressive embedded in interlaced is made by a process corresponding to telecine in the PAL standard [82]. Then each field i in each of these sequences is merged with its neighboring field $i + 1$. Q is made from the interlaced sequence with every frame having artifacts. Q' is made from the embedded progressive and will only have artifacts in every second frame, as every other second frame is a merge of a progressive original frame. Starting with n progressive frames we get n interlaced fields and $n - 1$ merged frames for building Q and $2n$ progressive-embedded-in-interlaced fields and thus $2n - 1$ frames for Q'.

Method One is detection by comparing Q and Q' distributions of short sequences to the archetype of progressive video, P. Thus, naturally, we in general expect $D_{KL}(P, Q)$ to be larger than $D_{KL}(P, Q')$, but in case of no or little difference they would approximately the same and make the correct classification as interlaced difficult.

Method Two is called Zigzag as it takes the distribution of every second frame, the subset $X = (1, 3, 5...)$, of a short sequence and compares it to the distribution of every other second frame, the subset $Y = (2, 4, 6...)$, of the same sequence. If the sequence is interlaced, $D_{KL}(Q_X, Q_Y)$ should be very small as both subsets have interlaced frames. But for progressive video embedded in interlaced, $D_{KL}(Q'_X, Q'_Y)$ should be large as you compare the distribution of the interlaced subset to the distribution of the progressive subset. In case there is no or very little motion in the sequence, $D_{KL}(Q'_X, Q'_Y)$ will be small and the correct classification as progressive will be hard to do.

6.2.6 Comparing the Two Approaches

D_{KL} is an asymmetric measure, making it well suited for the asymmetric data in method one, but not so good for the symmetric data ($[Q_X, Q_Y]$ and $[Q'_X, Q'_Y]$) in method two. As it turned out the use of D_{KL} in method two worked well in pratice, but the risk of problems could otherwise have been avoided by using a symmetric measure like the Jensen-Shannon divergence [41].

Building P for method one might give a very general distribution, maybe causing the difference between sequences to appear larger than the difference between interlaced and progressive. Method two does not have this problem as it measures locally on a given sequence, which then on the other hand could cause a loss of generality and uniformity over sequences.

Both methods fails to distinguish between the two scan formats in sequence parts without any motion. We do not consider this to be a problem, as this kind of video is also where a good motion adaptive or motion compensated deinterlacer will not harm a progressive sequence, just as frame merging intended to rejoin the two fields making up a progressive frame will not deteriorate interlaced video when there is no motion present – it will actually produce the best deinterlacing possible. Motion is *the* source of difference between interlaced and progressive video.

6.3 Other Work

The subject of scan format detection seems to have limited focus in academia, but it is a key element in actually building a video converter as can be seen in the patents [36] and [72]. Some industrial research has made it into academia, as can be seen in the papers [29] and [91]. They both use motion vector based film mode detection. None of the papers give any test results stating the quality of the methods.

A major reason for the lack of interest in scan format detection in academia is that with NTSC (the interlaced broadcast standard in USA and most of Asia) a 3:2 pull-down is used for telecine leading to a given cadence at which a number of field will be shown twice (see [72], [82]) simplifying the matter significantly [72]. The simplification does not apply to PAL telecine and the presence of noise will also complicate the NTSC case.

Nobody seems to have applied image geometry to the problem before us.

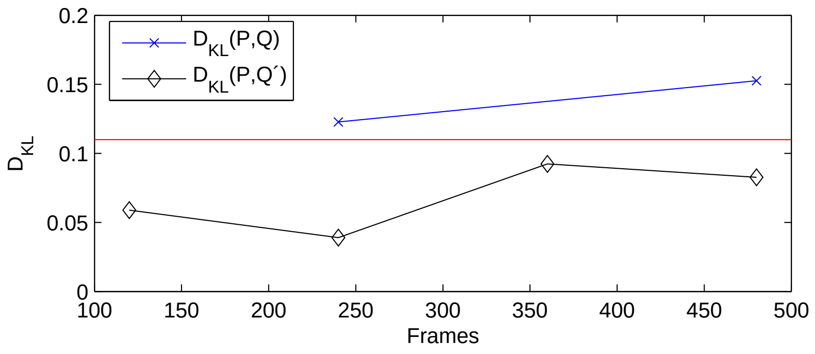


Figure 6.2: Method one: Threshold (horizontal line) in 'leave one out' test.

6.4 Results

For the testing we have used 8-bit gray-scale video corresponding to the luminance component of almost any TV or video signal. We have taken single chapters of 6,000-12,000 frames each from five different movies on DVD. They are processed in *chunks* of 480 frames each, the chunks subdivided into *bites* of 10-160 frames. The curvature is computed at different fixed *scales* in scale-space. If the *ratio* between extrema[1] values in D_{KL} for interlaced and progressive is larger than 1, then a *gap* exists and a *threshold* can be set to determine the scan format (see Figure 6.4). In classification terms that is: We have have to be able to draw a (1D) decision boundary that 100% (or as close to as possible) separates the two. The correctness can be measured by *recall* = correct/(correct + missed).

6.4.1 Initial Testing

Following the philosophy of keeping it simple, we started by doing some small tests. First we took two 40 frame bites (denoted a and b) from movie A and did a comparison of Q and Q' with P, first on a and then on b. Initial tests at scale 1.0 show that the $D_{KL}(P,Q)$ on both were a factor of four bigger than the $D_{KL}(P,Q')$ (ratio 4:1) thereby proving that interlacing introduces a difference in curvature distributions. Unfortunately the difference between the two different sequences, a and b, measured as $D_{KL}(Q_a, Q_b)$ and $D_{KL}(Q'_a, Q'_b)$ are a lot larger than the difference internally in each sequence between interlaced and progressive. This indicated that the scale might be wrong and that we would have to limit the measures to regions where the difference between interlaced and progressive is large, namely at edges.

Lowering the scale helped, but it was using edge detection and limiting ourselves to measuring curvature at pixels marked as edges that made it possible to separate interlaced and progressive at scale 0.5 (using a P made from both a and b). This showed that distribution of curvature at edges can be used to detect the scan format of a video sequence.

[1]The minimum of the top one and the maximum of the bottom one as in any of the figures in this section.

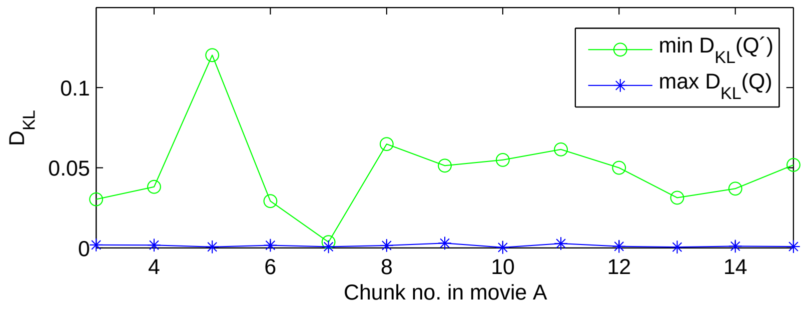


Figure 6.3: Method two: min. $D_{KL}(Q')$ and max. $D_{KL}(Q)$ for each chunk in movie A. A threshold would appear as a horizontal line in the plot. As can be seen, chunk 7 causes some problems.

6.4.2 Method One – Comparison with P

The last result presented above was promising and then we wanted to determine if it could be generalized to larger data sets and whether a general threshold to separate interlaced and progressive video could be set using method one. We use 8000 original progressive frames from movie A to build the distribution P at scales 0.2, 0.3, 0.5 and 1.1. Then we measured $D_{KL}(P, Q)$ and $D_{KL}(P, Q')$ using bites of 20, 40, 80, 160 and 240 frames.

First we did a 'leave one out' test by taking one chunk from movie A not used in building P. At bite length 240 and scales 0.5, 0.3 and 0.2, a narrow gap in which to set a threshold was present (see Figure 6.2). The ratios between extrema in D_{KL} were 1.33:1, 1.26:1 and 1.04:1 at the three scale respectively.

As a next step, measuring of D_{KL} on five chunks from the 8000 frames used to build P was done. Gaps were obtained for three of the chunks at low scales and for long bites. But the gaps were at different D_{KL}-values such that no common threshold could be set. Trying to use only the frames in Q' that are progressive did not help either.

To conclude, using method one – comparison to P – leaves the problem of separating interlaced and progressive unsolved.

6.4.3 Method Two – The Zigzag Solution

Movie A. Initial testing for method two was also done on the two bites, a and b, from movie A. using scale 0.5 and edge detection separation ratios of 439:1 and 52:1 between Q and Q' for each of the two bites where obtained comparing the worst D_{KL} values for Q and Q' respectively. As D_{KL} is asymmetric we get two D_{KL} measures for both interlaced and progressive as seen in Figure 6.5. All ratios will be given using the worst of the two choices of D_{KL}. As seen in Figure 6.5 the curves for the best and worse seem to meet whenever the gap between interlaced and progressive narrows.

Increasing the size of the test, a chunk of movie A was tested at scale 0.5 in bites of 40 frames and gave a ratio of separation of 6:1 for the full chunk. Lowering the scale (0.4, 0.3, 0.2 and 0.1) gave better ratios, the best being 10:1 at scale 0.3. Higher scales (0.9 and 1.1) gave no separation.

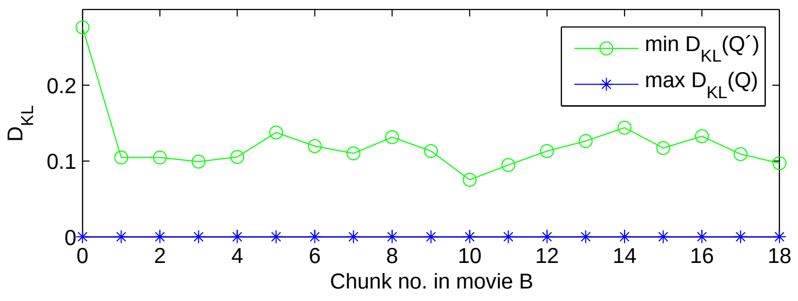


Figure 6.4: Method two: Excellent separation in movie B, which gives a rather free choice of the threshold value.

The effect of using different bite lengths was tested on the same chunk using scale 0.3. For the bite lengths 10, 20, 40, 50, 80 and 100 separation ratios were < 1, 2, 10, 7, 29 and 27. So the longer the bite, the better the separation – as expected.

We continued by testing the scales 0.5, 0.4, 0.3 and 0.2 at bite lengths 20, 40, 60 and 80 on two more chunks. From these tests scales 0.2 and 0.3 seemed the best with bite length 80. On the remaining 13 chunks from movie A processed at bite length 80, scale 0.2 performed better than scale 0.3 at the crucial parts where the gap between interlaced and progressive is small (Figure 6.5). Chunk 7 (Figure 6.3) makes it impossible to set a global threshold. As Figure 6.3 shows, changing the bite length to 160 eliminates this problem, allowing a threshold in D_{KL} to be set between 0.0030 and 0.0036.

Movie B. 19 chunks were tested and as Figure 6.4 shows we get an excellent separation at scale 0.2 and bite length 80 and the threshold can be set in the interval 0.00017 to 0.075. The good results for movie B could be caused by the fact, that the test sequence is set in daylight whereas the one from movie A is set at nighttime. But it is actually only for chunk 7 where the camera is stationary that movie A causes critical problems and thus we could used bite length 80 without deteriorating the quality of movie A by wrong processing.

Movie C consists of 12 chunks yielding an interval for thresholding ranging from 0.0027 to 0.0062 at scale 0.2 with bite length 80. Figure 6.5 illustrates how it is parts with a stationary camera (and only little object motion) that causes low values in D_{KL} for Q'. Thus wrong processing due to wrong classification would not cause a major loss of quality as problems mainly occur in bites with little or no motion.

Movie D. 22 chunks were tested at scale 0.2 and bite length 160 and gave the interval 0.0020 to 0.059 for thresholding, except for one 160 frame bite, which gave a unexplainable bump for the interlaced Q with a D_{KL} value of 0.0036. By visual inspection the bite did not distinguish itself in any way from its neighboring bites.

Using a threshold in the interval 0.0030 to 0.0036 would so far misclassify nothing progressive and only 160 interlaced fields, giving an interlaced recall of $31520/(31520+160) = 0.9949$. A way of getting a recall of 1 could be detecting cuts (like in [104]) and only allow changes between the scan formats when a cut within the bite is also detected as a change from interlaced to progressive would

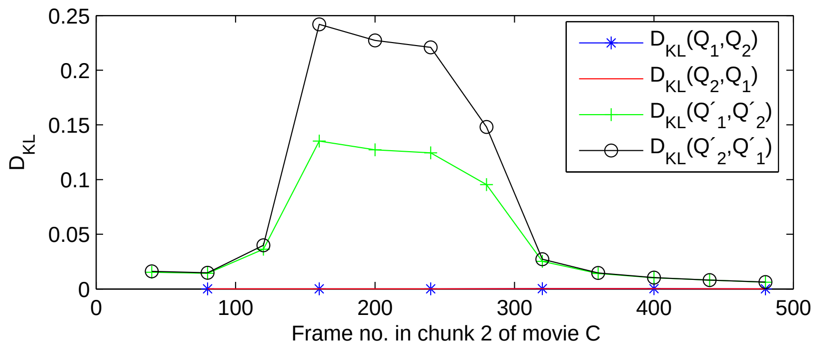


Figure 6.5: Method two: Parts with stationary camera corresponds exactly to parts with small differences between D_{KL} of interlaced and D_{KL} of progressive.

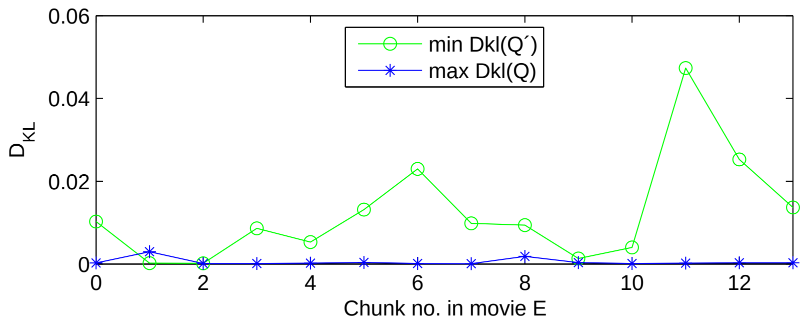


Figure 6.6: Method two: Problems in movie E, all though not as bad as this figure implies.

require a cut or possibly some type of softer transition. If the change is not a cut, one could then switch after a certain number of bites (or chunks) indicated a change.

Movie E. All tests so far has been conducted on natural image sequences, that is camera recordings of the real world, but movie E is computer animated and thus might give different results.

And so it did: Some of the 14 chunks processed at scale 0.2 and bite length 160 gave rise to problems as can be seen in Figure 6.6. However, of the total 84 bites of Q' in movie E, only six gave too low a D_{KL} to be classified correctly as progressive with a threshold between 0.0030 and 0.0036, yielding a recall for progressive detection in this sequence of 0.9286. Four of the troublemakers are in stationary parts of the main titles in chunks 1 and 2 (Figure 6.6) and the remaining two are in a part of chunk 9 where the camera is 100% stationary – as it can only be in computer animated films – and this part is also very dark, meaning that a wrongful deinterlacing would do no harm. At bite length 80 frames the six errors persists and, of course, doubles in numbers. Also new problems appear in seven bites at other places, but all in similar harmless scenes as for the previous ones mentioned.

6.5 Conclusion

Two methods to detect scan format has been set forth, only one of them solving the problem satisfyingly, namely method two – Zigzag. We recommend using scale 0.2 with a bite length of 80-160 frames. At these settings method two detects the correct scan format with recall 0.9875 for progressive and 0.9958 for interlaced. The interlaced miss of one bite in movie D is inexplicable. The progressive misses in movie E are all in parts with a stationary camera, little or no object motion and low-key lighting. In such scenes a wrong detection will not lead to significant creation of artifacts. Our method has sufficiently low complexity to be implemented in real-time hardware/software and thus used in a video converter.

6.6 Future Work

Some further work could be done to improve our scan format detector.

We have not tested material where each frame is a mix of the two scan formats, e.g. interlaced video with progressive graphics (news, MTV, etc.), film source TV broadcasts with interlaced generated subtitles, or some other mix. In these cases the gap between the scan formats narrows and some segmentation of the image plane is most likely needed to solve these problems properly. But in some cases (e.g. stationary progressive graphics in interlaced video) our algorithm combined with a good motion adaptive or motion compensated compensated deinterlacer will most likely yield acceptable results.

Bite lengths of 80-160 frames corresponds to 1.6-3.2 seconds of switching time in PAL, which is clearly acceptable. Trimming our algorithm to use shorter bites will make switches between interlaced and progressive faster in programs mixing the formats inter-frame (documentaries and movie featurettes). One way of doing this could be combining edge detection with (simple) motion detection to get fewer but more significant data points for processing.

Chapter 7

Summary and Future Work

In this chapter we will summarize the main contributions of this thesis and discuss some important possibilities for future development of our methods. Most of the development possibilities are on improving the visual output quality, but the most interesting one is on speed-ups and it gives the main directions towards realtime implementations of variational upscaling methods. But first a summary of the work already done.

7.1 Summary of Contributions

The main contribution of this thesis has been to investigate the usability of variational methods in an general applicable upscaling video processor. Bayesian inference has been used to impose regularity of the ill-posed problems of image sequence restoration and enhancement. We have not tested a wide range of different probability functions, but focussed on the use of total variation, which composes a nice balance between on one side mathematical tractability and computational cost and on the other side fidelity in modelling and produced output quality. Since our Bayesian framework and the derivation of variational methods from it was used earlier for inpainting by Lauze and Nielsen in [65], and since variational optical flow methods are well-known in literature, it has mainly been the application and adaption of methods to the upscaling problems that has been new. We will now summarize the contributions in detail chapter by chapter.

Chapter 2: Background

We have discussed how properties of the human visual system dictate the need of high quality upscaling to improve the viewing experience (if no high definition source material is available). High quality image sequences enables the viewer to focus on the journalistic, artistic and entertainment aspects of watching motion pictures, video and television to a degree where technology becomes invisible.

The use of state of the art variational optical flow methods in motion compensated upscaling is introduced, and the optical flow methods are integrated in our Bayesian framework for simultaneous optical flow and intensity calculations, from which a variational energy minimization formulation is derived.

Chapter 3: Deinterlacing

We show that no motion adaptive deinterlacer including our variational one can solve The Interlacing Problem of highly detailed regions in motion. The Interlacing Problem can only be solved by motion compensated methods gathering information temporally along the optical flow field.

We analyze and discuss the problem of computing optical flows on interlaced video for motion compensated deinterlacing. We chose a strategy computing the flow from the interlaced data alone and use that as input for variational motion compensated deinterlacing.

The Interlacing Problem is in all test cases solved by our variational motion compensated deinterlacer, which yields high quality results with hardly any artifacts present and close to the ground truth. Still, there is room for improvements in quality, the biggest improvement most likely to come from doing simultaneous calculation of flow and intensities.

Chapter 4: Video Super Resolution

The variational motion compensated video super resolution method presented in Chapter 4 does do simultaneous flow and intensity calculations, producing better results than our earlier non-simultaneous variational video super resolution method presented in [54]. Since the latter non-simultaneous method do not compute precise high resolution flow, it does not benefit from temporal information to the same degree as the simultaneous method.

As data term in our model we use the super resolution constraint, which is derived from the image acquisition model, and controls the diffusion process by projecting the suggested energy updates back onto the true solution hyperplane.

Our simultaneous video super resolution method is in terms of output quality clearly better than bicubic and bilinear interpolation (the latter widely used in video processing devices today) and is also shown to outperform the super resolution methods of highly expensive film post production and editing systems (but only on one test example so far).

There are super resolution methods in literature that are likely to perform better than our simultaneous variational video super resolution, but due to limitations in what types of depicted objects (e.g. faces only) or flows (e.g. parametric flow only) the applied models allow – or the need for multiple camera recordings of the scenes – these methods are not applicable to general video with arbitrary (natural) content and motion as our method is.

Chapter 5: Temporal Super Resolution

The presented variational motion compensated temporal super resolution method derived from our Bayesian framework simultaneously computes flows and intensities in a multiresolution setting. No other TSR method has computed flow and intensities of nonexisting frames simultaneously, but just estimated flow fields in new frames as a preprocessing step to the intensity calculations.

In Chapter 5 we went into more details about the human visual system than in the background chapter (Chapter 2) to find out what requirements are put on frame rates in modern cinemas and video display systems.

Derived from our variational temporal super resolution framework two versions of variational frame rate doubling are implemented and tested. In the first version we used the gradient constancy assumption in the flow energy minimization as done by Brox *et al.* in [9], but to get a theoretically consistent and less complex algorithm, we leave out the GCA in the other version. The latter is expected to give smoother and less precise flows.

We do not always create perfect new frames, still both versions of variational motion compensated temporal super resolution do produce high quality 50 fps video from 25 fps video without noticeable artifacts during video playback and thus reestablish the pi-effect and the illusion of motion pictures for the problem case of high contrast edges in motion. The version using GCA in the flow energy minimization does give a sharper output in case of object motion, but the difference is not perceived when playing back the frame doubled results as video.

Although we have only implemented and tested a frame rate doubler, implementation of a generic frame rate converter from our variational formulation of the temporal super resolution problem is straightforward.

To truly prove the quality of our results, both for TSR and our two other upscaling methods, we need to compare our outputs with those obtained using other motion compensated upscaling methods. The problems of doing benchmarks is discussed in Chapter 5.

We also discuss the general problem of handling certain types of complex flows with our current implementation of variational optical flow. Suggestions on how to solve the problems are given.

Chapter 6: Detecting Interlaced or Progressive Source of Video

We have designed, implemented and tested of a method for detecting input scan formats. This is a crucial preprocessing step to any video upscaling system as it decides on whether or not to deinterlace the input video signal. Wrong interlaced/progressive classification of the input could have severe consequences to the final output quality of the upscaling system. Our method was shown to classify correctly in more than 98% of all cases, only failing in cases where the image content and motion were of a character that would not lead to bad output quality in case of wrong processing.

7.2 Possible Future Developments

The variational methods for deinterlacing, video super resolution and temporal super resolution presented in this thesis all produce high quality results and are ready for 'real' use in software or hardware products. There is however some remaining problems, that we would like to (try to) solve in the future. We devote separate sections to each problem, but first a short introduction of each of them.

- **The Modelling Problem.** Currently we use total variation as the probability function in all terms of our framework and although it is a good model of image sequences we should be able to do better.

- **The Flow Problem.** We have found that the use of state of the art variational optical flow methods (see [12] by Bruhn *et al.*), helps produce high quality motion compensated upscaling results, but the flow computation is in our opinion the element of variational upscaling methods with the greatest potential for improvements.

- **Speedup.** It is no secret that our current implementations are slow and to achieve realtime running times speedups are required.
- **Integrated Upscaling Systems.** We now do upscaling in three separate steps even though all three build on the same basic technology. When two or all three steps are applied to a video throughput integrations are possible.
- **Streamed Processing.** When taking our methods out in the real world, we are no longer processing short test sequences but have to process continuous streams of video throughput.
- **Benchmarking.** To give non-repudiable documentation of the performance of any method for video processing, one needs to benchmark against what is otherwise considered state of the art within a field. We will not discuss this topic further as it was thoroughly covered in Section 5.2.8 of the temporal super resolution chapter.

7.2.1 The Modelling Problem: Priors and Data Terms

Total variation is widely used in image (sequence) content modelling; its preference for smooth regions – a property shared with the Gaussian – combined with its preservation of edges is closely related to how the lower level vision works. Humans lower level vision perceives mainly edges and then fill in the colors of smooth regions from there – just as information is transported along and away but not across edges in total variation diffusion. Still total variation is not state of the art in salient image content preservation as e.g. structure tensor based models have shown higher degrees of edge preservation and better handling of corners. The problem of using structure tensors is their high computational complexity, but the same was said about the nonlinear total variation back when linear Gaussians where the most widely applied model. Thus one day variational, or more broadly PDE-based, video upscaling is likely to be done using structure tensors. Structure tensors have been used extensively in 2D image plane space (in e.g. the works by Roussos/Maragos [86] and Tschumperlé/Deriche [103] but spatiotemporal structure tensors has been introduced, e.g. in optical flow computations by Brox *et al.* in [10].

A family of models that might be tried as an alternative to structure tensors is the family of models learned from doing statistics of natural images (and image sequences). By building distributions from a large data set of natural images, models that at a glance look like total variation with high kurtosis at the mean and heavy tails, are typically obtained, an ikonic example being the work by Zhu and Mumford in [110]. These models are different from total variation, but the differences are subtle and the resulting images from different processing with the Grade algorithm from [110] look very cartoon-like and could from their appearance be guessed to be the results of diffusions using total variation. But the Grade algorithms is known to be able to build structures total variation is unable to build, thus making models based on statistically learning interesting for video upscaling. An overview of the field of building models from statistics of natural images can be found in [96] wherein Srivastava *et al.* conclude that with the sample based models of today, *"we are still quite far from a full probability model"* (p. 29). The paper was published in 2003 but the argument still holds. What would really be of interest to us is any extension of the models reviewed in [96] to also model images sequences, that is adding the temporal coherence and correlation to the already existing spatial model. Some examples of work pointing in this direction are [39] by Fitzgibbon focussing on image registration in time sequences and [30] by Doretto *et al.* on dynamic (in time) textures, which brings into focus the whole field of texture modelling to create details not possible with probability models such as total variation. Also of interest are the disciplines of exemplar-based inpainting (see for instance [26] by Criminisi *et al.*) and patch-based image modelling and reconstruction, see for instance the paper [44] by Griffin and Lillholm and references therein.

The idea of reconstructing, repairing or enhancing images from the (visually) prominent features in the image is well-known (see for instance [68] by Lillholm *et al.*). Features (edges, ridges, corners etc.) are in image analysis often seen as carriers of salient image information in parallel to how the lower level vision in humans perceive visual input. Small patches can also be considered to carry salient image information and thus be a good representation of images. These patches are often learned as in e.g. [40] by Freeman *et al.* and often they are found to have structures similar to the salient features. The set of pathes representing the features salient to the lower level human vision is know as textons (see e.g. [44]). 3D spatiotemporal patches as representation of image sequences are presented under the name of video epitomes (as already discussed in Chapter 5).

Before introducing structure tensors, statistically learned priors or any other modelling to video upscaling, there is a more likely addition to be tested thoroughly, and that is use of the gradient constancy assumption in the intensity part of our energy minimization. Its use has been rejected by logical arguments and the fact that it is computationally heavy, but it needs to be given the chance to prove its worth before being discarded permanently.

7.2.2 Improving Flow Quality

As indicated several times throughout this thesis there is a need of improved (variational) optical flow computations on real world image sequences to lift motion compensated upscaling to the next level in terms of perceived output quality. Results are already good, but more reliable and precise flows would lift the quality further. Additional improvements in quality could be found in increasing the robustness against unreliable flow vectors in the motion compensated part of our upscaling algorithms. A discussion of the main problems with flow computations and handling plus suggestion on how to solve them has been given in Section 5.5.1 of the temporal super resolution chapter.

7.2.3 Speedups

It is no secret that the code used to generate the results of this thesis is not exactly fast, still we claim to be able to do realtime applications at reasonable costs. The running times reported on video super resolution in Section 4.4.5 are representative for all three upscalings. We see that it is the initial low resolution flow computations in the multiresolution pyramid that takes up most of the running time, but we know this step can be run in realtime for simpler variational flows (again we refer to the paper [11] by Bruhn *et al.*). To get a precise but computationally cheaper flow calculation, one could run such a simpler (linear) variational optical flow method or a block matching algorithm to get an initial estimate of the flow and then refine it with the computationally more expensive and accurate method on only a few of the finest scale levels of the pyramid used in our current upscalers. Alternatively one could also run the cheaper method and then only do very few iterations of the expensive method on each level of the pyramid.

No matter if we bring in an additional flow algorithm or not, we can also make our current variational flow method faster. And the same goes for the intensity calculations. We have used Gauss-Seidel solvers in our implementations, but switching to solving the systems using the faster successive over-relaxation (SOR) is just a matter of adding a very few lines of code. However, it will take quite some time to test and tune the altered code and the new SOR weight parameter. Further speedups could be gained switching to multigrid solvers, which have been used to successively speed up variational flow calculations, e.g. in [11], but this requires a larger and not straightforward restructuring of our implementations. General optimizations of the code the use of faster solvers will be beneficial to the performance of both software and hardware implementations. Our variational methods runs the same filter on all pixels, making them highly parallelizable, which is an advantage in hardware implementations but also in software implementations since the growth in CPU speed is being replaced by a similar growth in the number of cores in CPUs.[1] Still we do not find it highly likely that we will reach realtime performance in software in the near future without using dedicated and specially developed hardware. We do not consider the high level variational parallelization presented by Kohlberger *et al.* in [62] as something we want to apply to our upscaling algorithms since it is not very efficient. We believe a direct parallelization splitting the frames into spatial regions with a small overlap between them and optimizing each region on its own processor with a shared memory for boundary data is the way to reach realtime performance in hardware. The hardware we have our minds set on using are HDTV FPGAs able to run just under 100 mathematical operations per pixel on a 1080×1920 HD sequence in realtime.

A very obvious way to gain a huge speedup is through integration of forward and backward flow calculations. So far we have ignored the fact that the forward flow from frame n to frame $n+1$ and the backward flow from frame $n+1$ to frame n should be practically the same, they are just the direction specific warps or mappings from one frame to the other. The two might not be exactly the same, which is mainly do to numerical imprecision and (dis)occlusions, but when one of them is done at any given level of the multiresolution pyramid, the reverse of it will with a very few extra iterations be the other, and we can then save a lot of processing time. Similar improvements are possible with warping from input sequence flow to output sequence flow in variational temporal super resolution.

We have so far processed all pixels equally, but as some types of image

[1]The classic Moore's law predicts a doubling in the number of transistors in CPUs every 18 months, but since cooling of single core processors is becoming a problem, the doubling in processing power every 18 months is now predicted to be obtained by growth in the number of cores in CPUs.

(sequence) content is very simple we could use very simple processing on these types of content. There is a tendency in our work as well as in many other works in the field of image (sequence) processing and analysis to focus on the difficult cases in a given data set, but for large portions (mainly the smooth regions) of the data very simple processing would do. The problem lies in identifying the troublesome regions; methods for edge detection, motion detection on interlaced (and progressive) video, general segmentation, etc. are not fail safe and often rather complex in them selves, and thus saving advanced and expensive processing might come at a high price of either other expensive processing or drops in output quality.

7.2.4 Towards an Integrated Variational Upscaling System

The complexity and running times of our algorithms in a system setup could be lowered by integrating deinterlacing, video super resolution and temporal super resolution into one variational upscaler. Although a complete integration, exists in theory as formulated by our Bayesian framework, we can only do little in practice to fully integrate our three components, but many minor integrations can be done.

An obvious place to integrate is in the filters applied: They are the same for all three algorithms except for the intensity data term (E_0/P_0) in each algorithm as we use the same regularization and flow data terms.[2] Therefore we can reuse (at least the design of) the filters.

Since the basic flow of a hardware upscaler is cascaded processing, one should of course reuse what can be reused, there is for instance no reason to recalculate the low resolution flows from scratch. If it has been computed for the deinterlacing step which is normally placed early in the processing pipeline, one can just reiterate the flow a few times before using it for VSR and/or TSR.

Integrating the flow calculations for deinterlacing and the full TSR part could also be done. After a downscaling of a factor of two in the height, the data we work on for DI and TSR (for the initial, original sequence flow) is the same, and thus most of the TSR calculations of flows and intensities could be done in parallel with multiresolution flow calculations for deinterlacing. Thus TSR should be place early in the pipeline.

In [92] Shechtman *et al.* presents a method for integrated spatiotemporal video SR (TSR integrated with VSR). Without going into details of how this methods is construed, its core idea is a multi-camera approach requiring that a) the recorded scene is planar with intraplane action only, or b) all the cameras are placed in practically the same position, both a) and b) giving an abundance of information easily realigned in a common spatiotemporal coordinate system. This, in combination with the following limitations in the modelling (e.g. no motion estimation making it motion adaptive spatiotemporal SR) makes their idea of how to integrate TSR and VSR non-transferable to our single-camera, general video problem. Logically, with just one camera available and using multiresolution for spatiotemporal super resolution, one would start by doing TSR and when reaching the finest spatial resolution (that of the input) switch to VSR with a low resolution flow already available. Doing VSR followed by TSR is also possible, but would mean some extra work traversing the pyramid twice (once

[2]The flow computation without GCA in TSR ignored for now.

for flow only calculations and then for both flow and intensity calculations).

Integrating deinterlacing and VSR is similar to integrating TSR and VSR with deinterlacing simply supplying the LR flow and intensity input to VSR. In some sense the deinterlacing is just the first level of VSR, but the 'up-and-down-bobbing' interlaced sampling grid complicates matters, making deinterlacing a bit more than 'just' VSR with a vertical magnification factor of two. (Doing VSR followed by deinterlacing would be plain stupid as VSR logically follows deinterlacing.) Integrating the interlaced sampling grid into the SR constraint is possible, although it would be more difficult do the back projection when lines of LR pixels are missing and one thus has to decide how to control the diffusion when there is no reliable anchor point to bind the new high resolution data to. The problem is basically that the subsampling of the interlaced grid is a spatiotemporal problem whereas the SR problem and our SR constraint derived from it are purely spatial.

Integrating either TSR or deinterlacing with VSR is difficult due to the fact that the spatiotemporal space is – as we have discussed several times – not just a 3D space, but a 2D + 1D space. (Also illustrating the problems of doing high quality deinterlacing as it is a spatiotemporal problem at birth.) On the other hand interlacing was in its days of glory a nice trick taking advantage of spatial and temporal characteristics of the human visual system. Truly integrating VSR with TSR and/or deinterlacing is a challenging and (thus) interesting problem, but in practice it would not change a lot: We already used the same filters, and the number of pixels needed to be processed in total stays the same.

Returning to integrating 'only' deinterlacing and VSR, we will always need at least two processing options in our system as we have two possible input types, interlaced and progressive. Either we will have to be able to do integrated or cascaded deinterlacing and VSR or pure VSR. From a practical and quality focussed point of view (overlooking for a moment the greater good of basic research), instead of trying to integrate the different types of upscaling, resources is better spend improving image (sequence) modelling. Increasing simultaneousness in intensity and flow calculations would also help, but as the multiresolution low resolution flow calculation in general is the heavy part, faster methods there would be more appreciated.

As we – hopefully – dig into the hardware development, more integration and speedup possibilities are likely to present them selves, but we might also have to compromise the output quality here and there to keep the cost of the product down, maybe produce both a basic and a high end version.

7.2.5 Streamed Video Processing: Temporal Sliding Windows

So far we have processed smaller sequences (5-20 frames) as single volumes as limited by PC memory. When our methods are to be applied to thousands and thousands of frames of video in a constant stream, some kind of sliding temporal window processing would be the obvious way to process the video. We could still process in separate volumes, but the sliding window has several advantages.

First, it is costly both in memory and cut detection to process scenes in one go and separating scenes internally might no be desirable due to boundary issues. For promotional reasons we did an experiment with our non-simultaneous VSR algorithm from [54] on two long sequences, one of 290 frame and one of 610 frames. They were processed in volumes of 14 and 16 frames respectively, but in both cases cut to a length of 10 frames to discard the purely spatially processed first and last frame (see discussion on VSR boundary issues in Chapter 4) and one or two of their neighboring frames to make 100% sure that we avoided any side effects from having temporally 'bad' information at the ends of the volumes. This is in terms of speed the downside of processing in separate volumes. On the positive side, we found in this experiment that our VSR algorithm handle the many cuts in the two test sequences just fine: Minor drops in sharpness was seen at cuts, but this was only noticeable when viewing stills, not during normal playback.

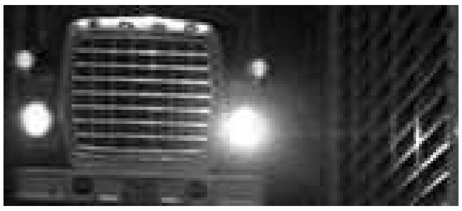
The second advantage of windowing is that we can warp information forward using the flows computed on the video stream, thus obtaining good initial approximations in each frame speeding up the processing. However boundaries meet when warping at (dis)occlusions, scene cuts and at spatial borders of the video frames should be handled carefully.

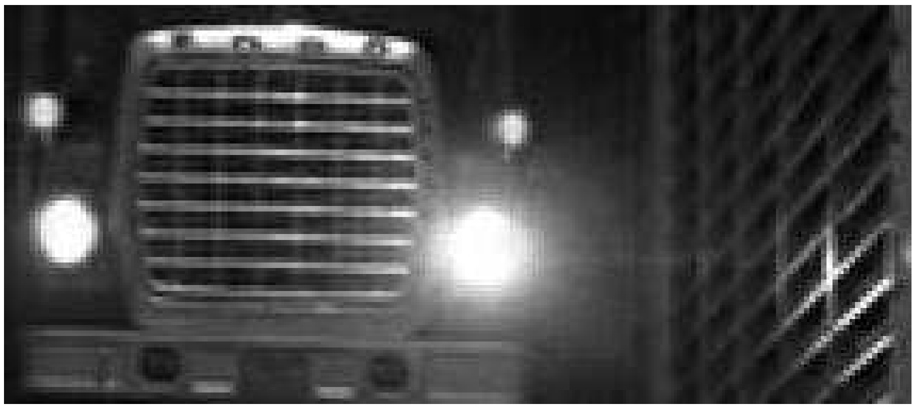
That's all Folks!

Appendix A

Large Size Video Super Resolution Figures

To give a better impression of the output quality obtained in video super resolution we give the result figures of Chapter 4 here at more appropriate sizes. In many of the figures in Chapter 4 a zoom percentage is given. It is the suggested zoom we have used here as far as the page layout allows for it. (Two empty pages are inserted amongst the figures to ensure that paired figures can be viewed on the same spread in case of two sided printing.) We still recommend electronic viewing as the printing process might smear and smooth out details in the figures.


(a) LR input (70 x 160 cutout)


(b) HR 2x2 initialization

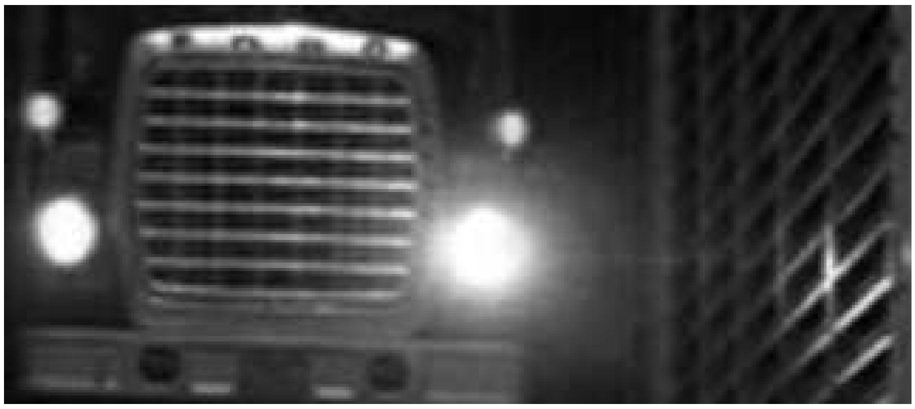
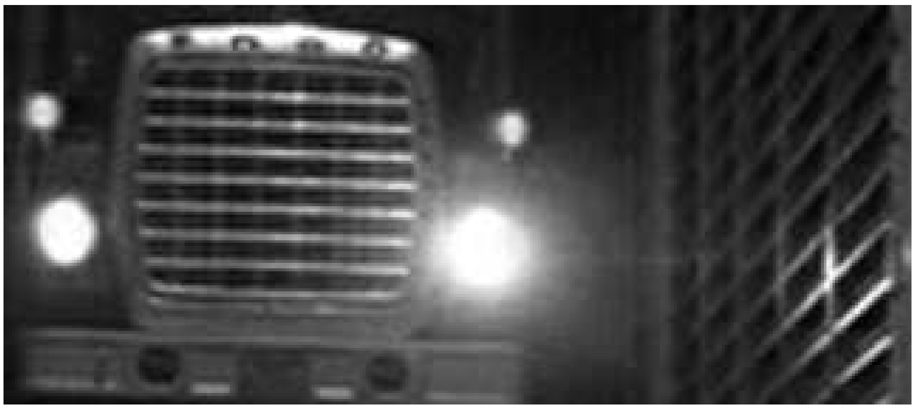
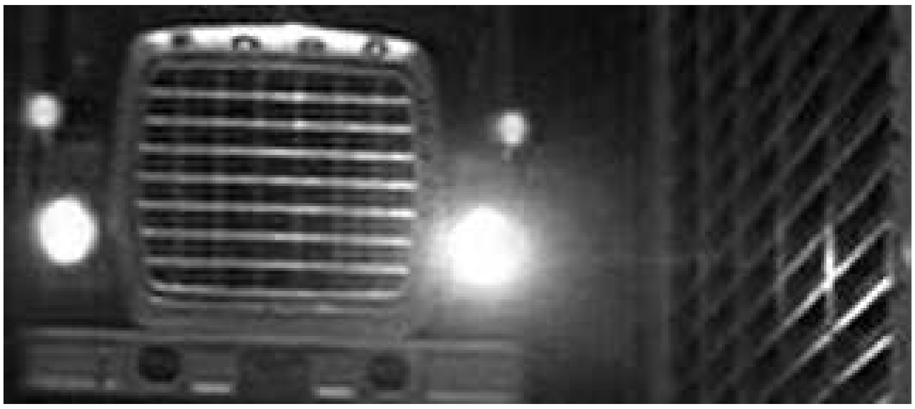

(c) Bilinear 2x2 SR

Figure A.1: 2x2 VSR on the sequence `Truck`. 200% zoom of Figure 4.4. Continued in Figure A.2.


(a) Bicubic 2x2 SR


(b) Nonsimultaneous 2x2 VSR from [54]

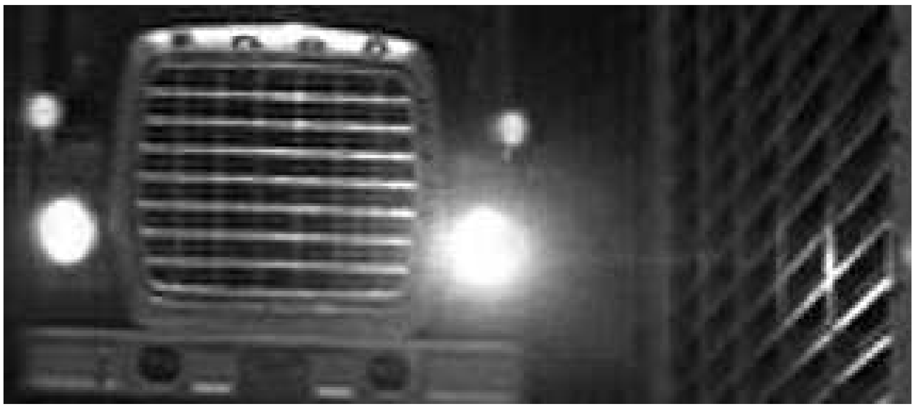
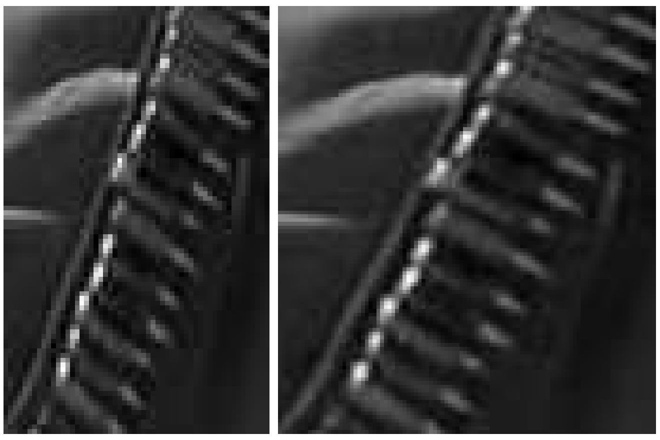
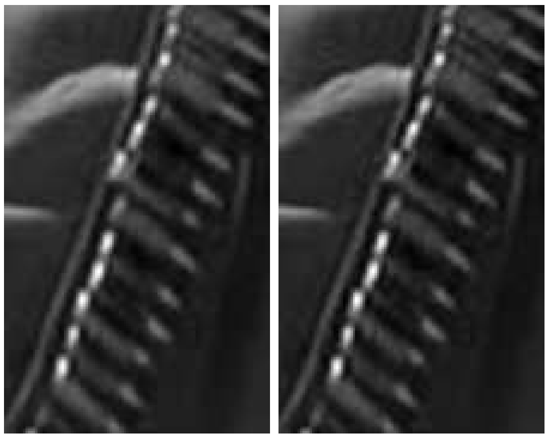

(c) Simultaneous 2x2 VSR (S-VSR)

Figure A.2: 2x2 VSR on the sequence `Truck`. 200% zoom of Figure 4.4. Continued from Figure A.1.


(a) LR (b) 720p S-VSR


(c) 2x2 Bilinear (d) 2x2 Bicubic

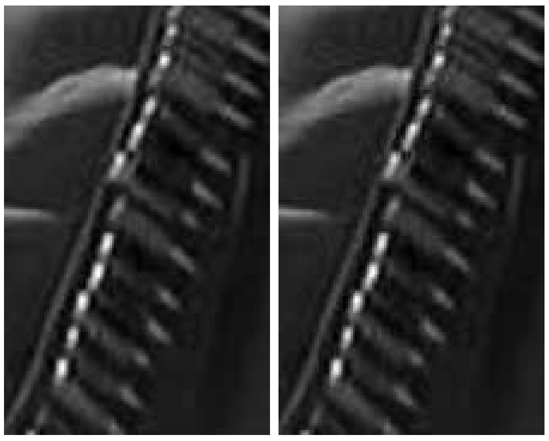
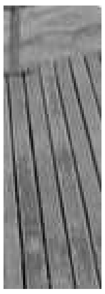
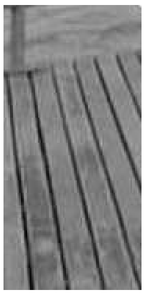
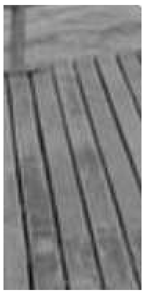
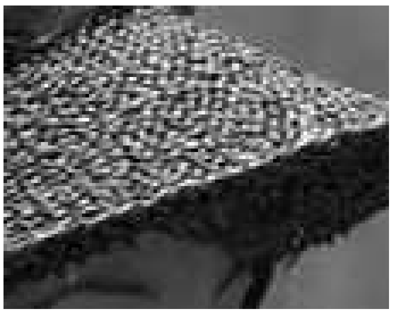

(e) 2x2 VSR [54] (f) 2x2 S-VSR

Figure A.3: VSR on the sequence `Bullets`. 200% zoom of Figure 4.6.


(a) LR


(b) S-VSR


(c) Bilinear


(d) LR

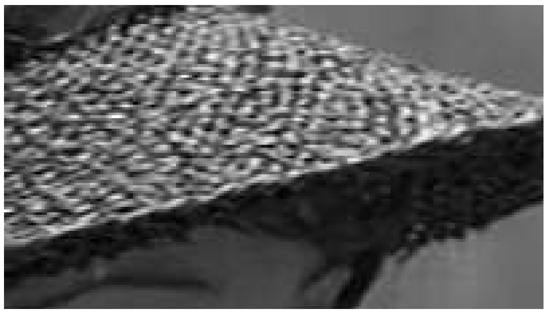
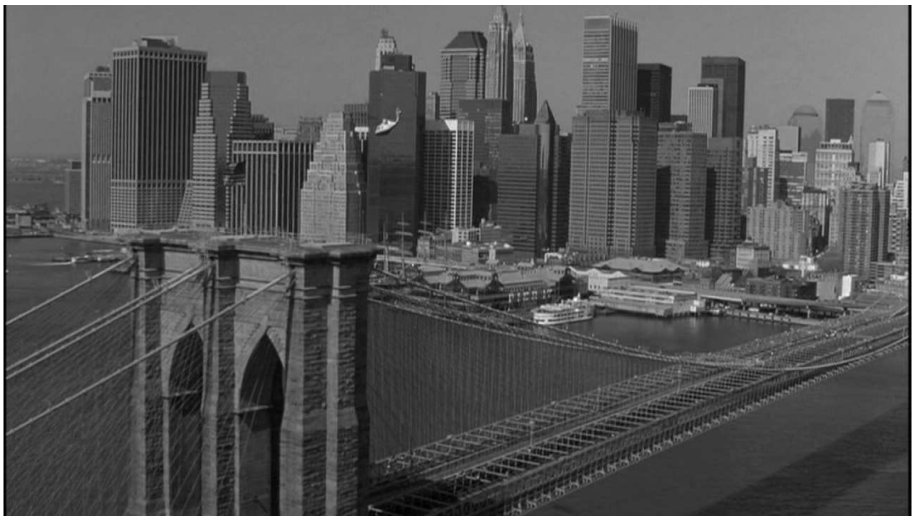

(e) 720p simultaneous VSR

Figure A.4: 720p VSR on the sequences `Boardwalk` and `Straw Hat`. 150% zoom of Figure 4.7.


(a) 720p simultaneous VSR

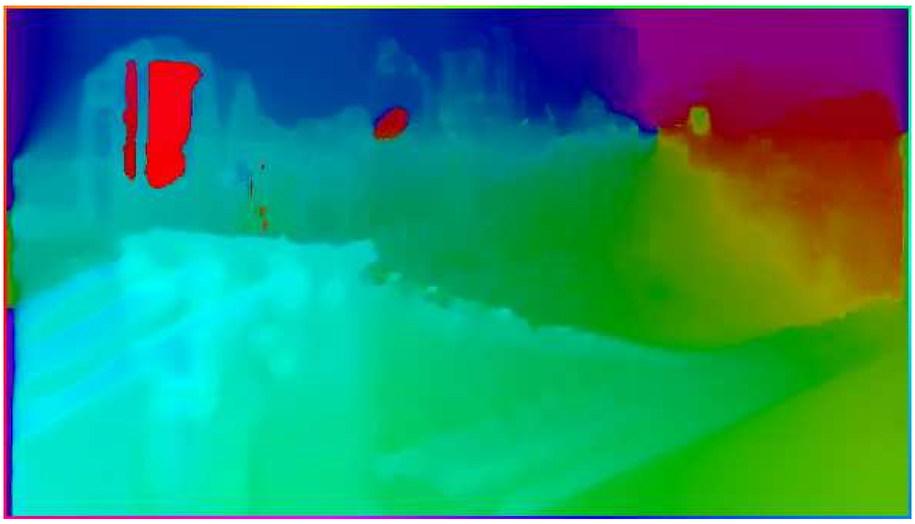
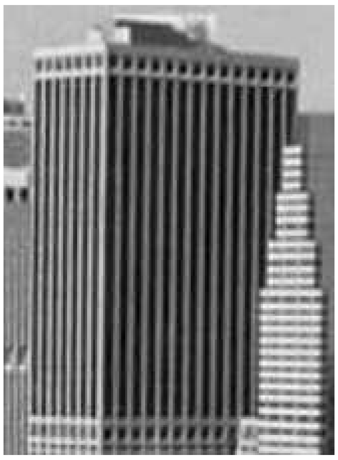
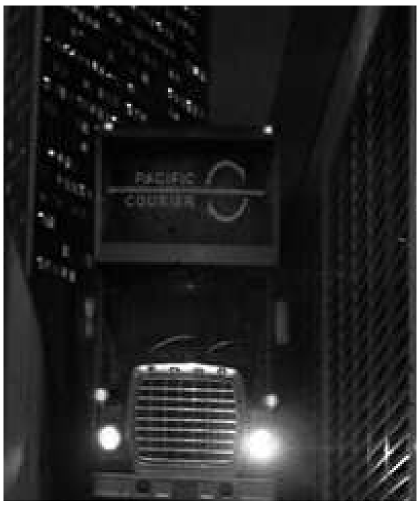

(b) Corresponding flow

Figure A.5: 720p VSR on the sequence `Manhattan Flyby`. 250% zoom of Figures 4.8(a) and 4.8(b).

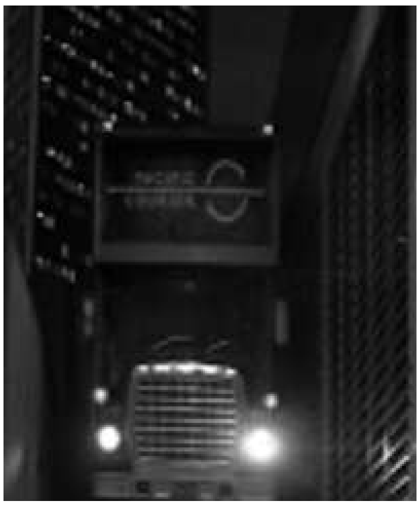


(a) Flow error handling

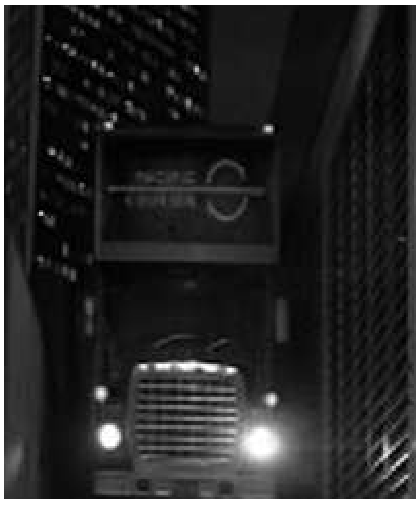
Figure A.6: 720p VSR on the sequence `Manhattan Flyby`. 200% zoom of Figure 4.8(c).



(a) Original

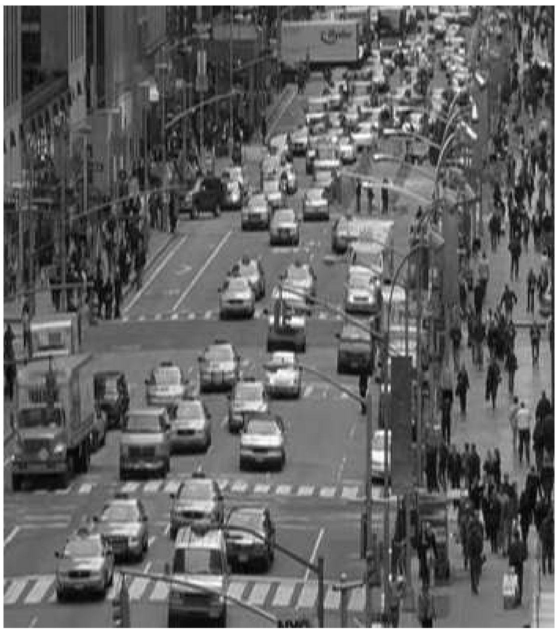


(b) 2x2 bicubic



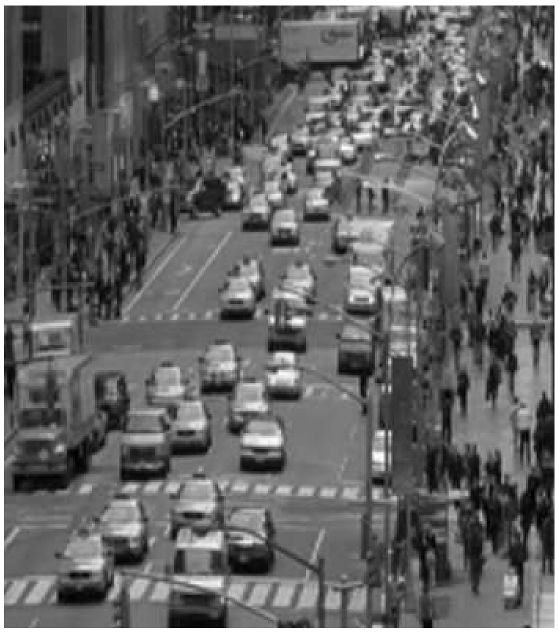
(c) 2x2 simultaneous VSR

Figure A.7: Down- and up-scaling of the sequence `Truck`. 150% zoom of Figures 4.10(a)-(c).



(a) Original

Figure A.8: Down- and up-scaling of the sequence `Street`. 200% zoom of Figure 4.10(d).


(a) 2x2 bicubic

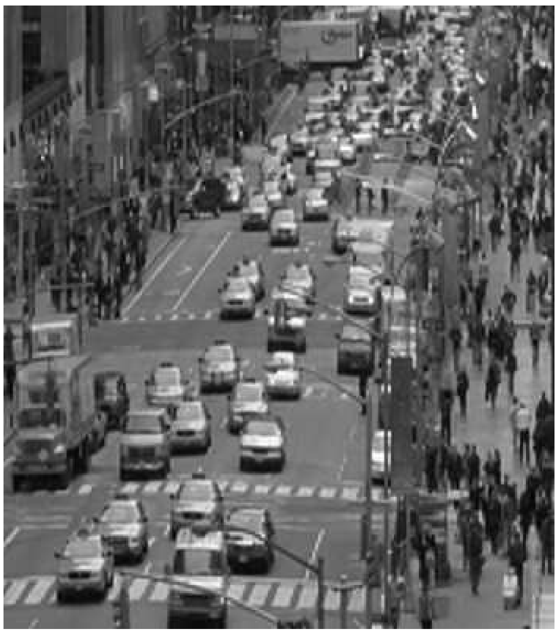
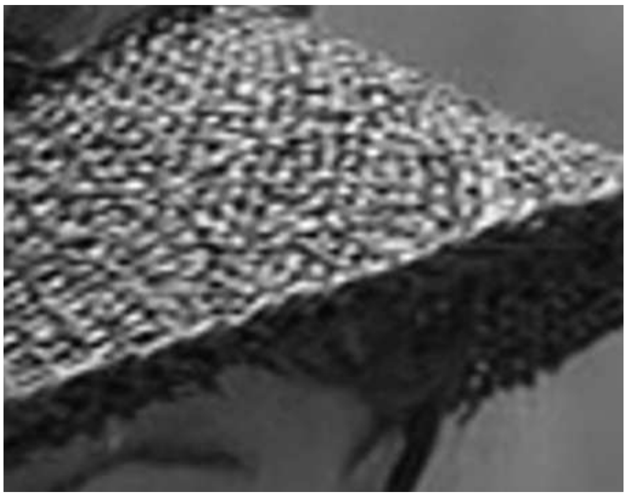

(b) 2x2 simultaneous VSR

Figure A.9: Down- and up-scaling of the sequence Street. 200% zoom of Figures 4.10(e) and 4.10(f).


(a) 4x4 bicubic

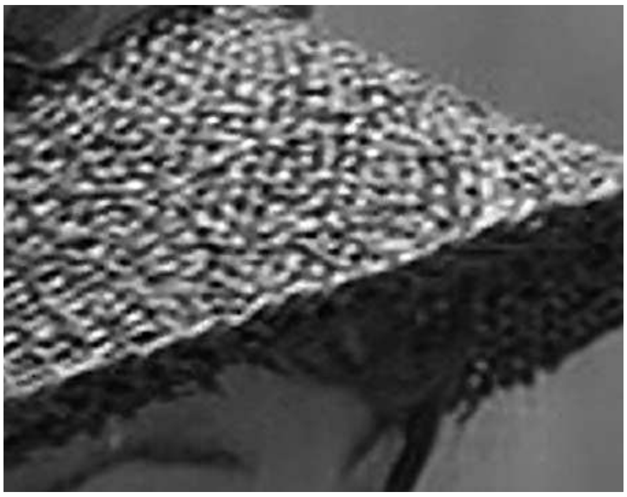
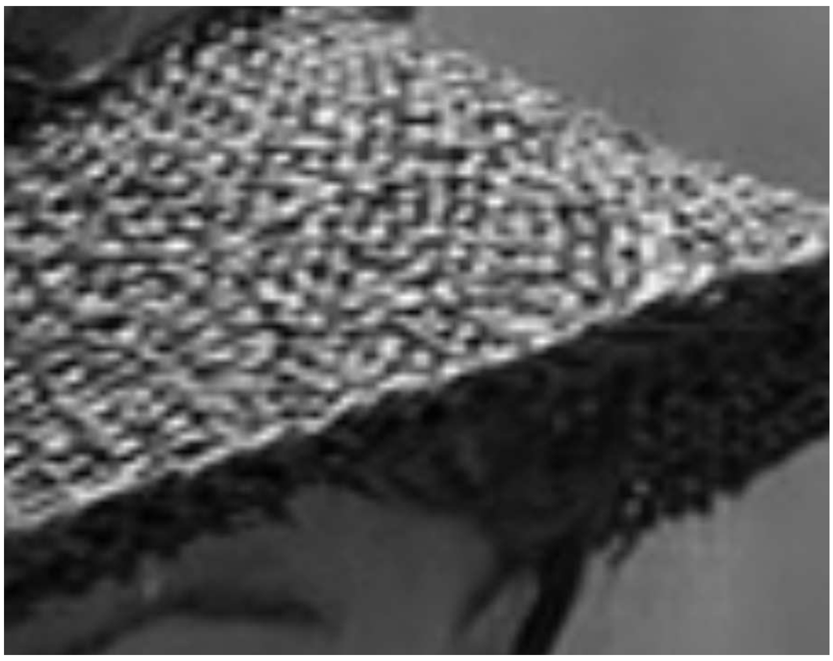

(b) 4x4 S-VSR

Figure A.10: 4x4 VSR on `Straw Hat`. 300% zoom of Figures 4.11(a) and 4.11(b).


(a) 8x8 bilinear

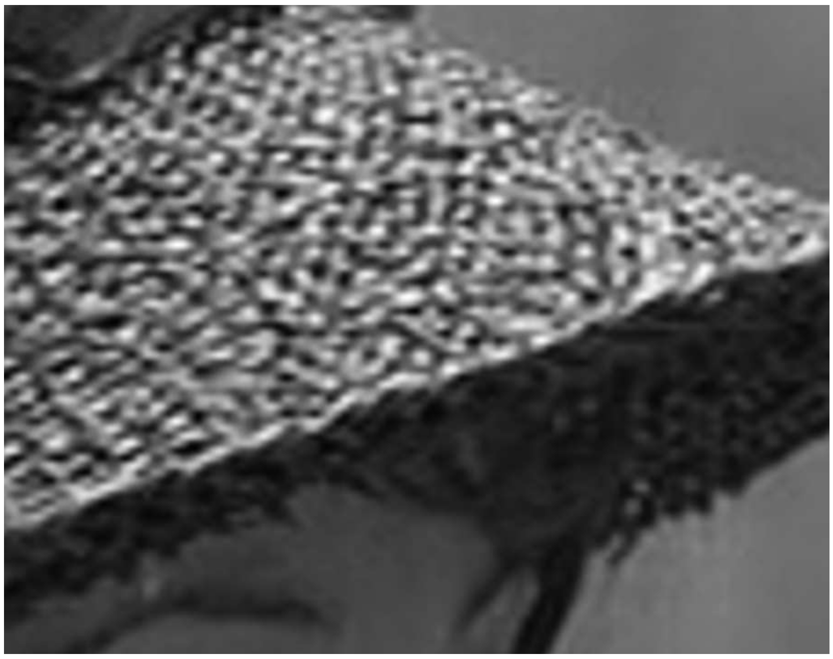
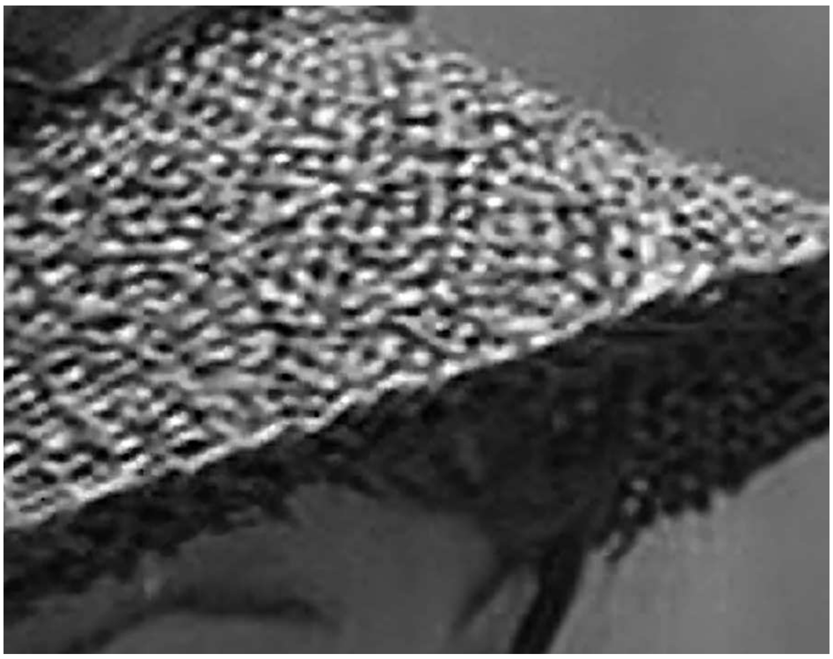
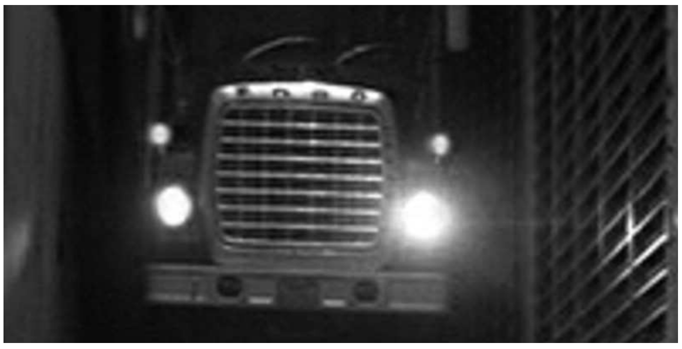

(b) 8x8 bicubic

Figure A.11: 8x8 VSR on `Straw Hat`. 200% zoom of Figures 4.11(c) and 4.11(d).



(a) 8x8 S-VSR

Figure A.12: 8x8 VSR on `Straw Hat`. 200% zoom of Figure 4.11(e).


(a) 4x4 bicubic

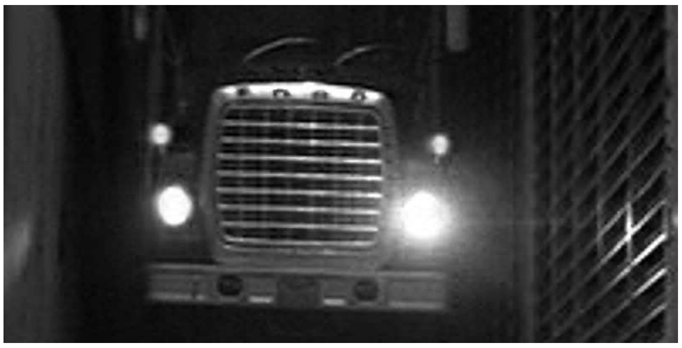
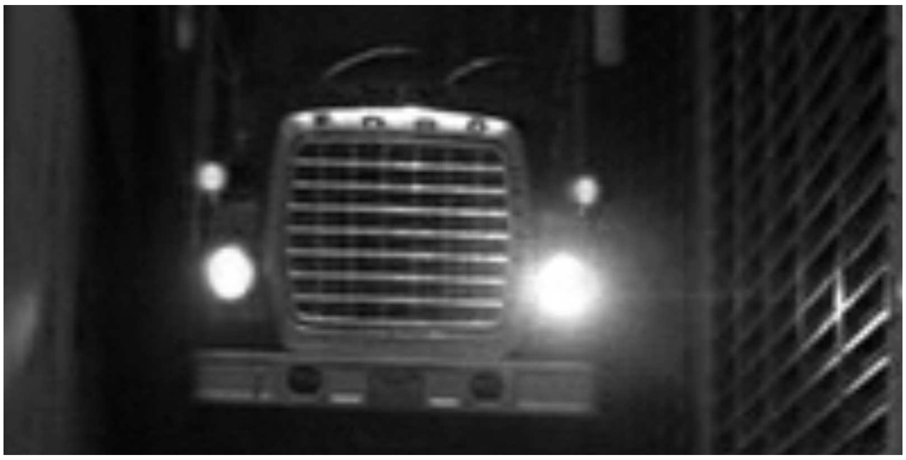
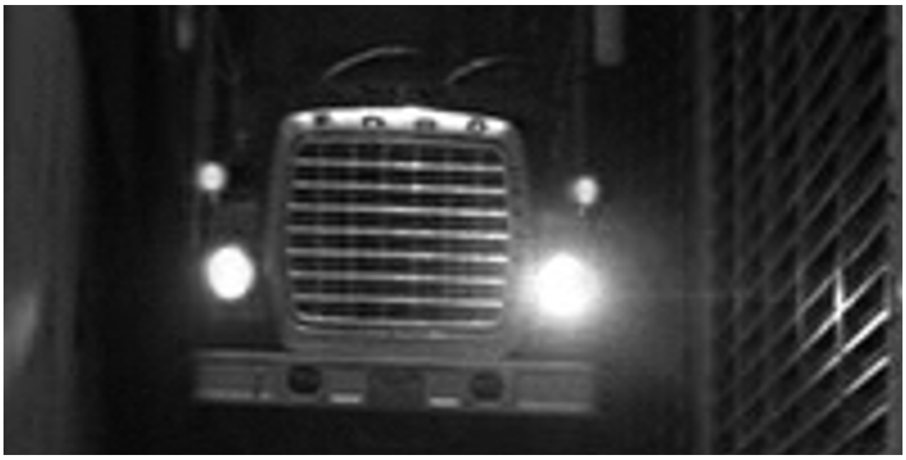

(b) 4x4 S-VSR

Figure A.13: 4x4 VSR on `Truck`. 300% zoom of Figures 4.12(a) and 4.12(b).


(a) 8x8 bilinear


(b) 8x8 bicubic

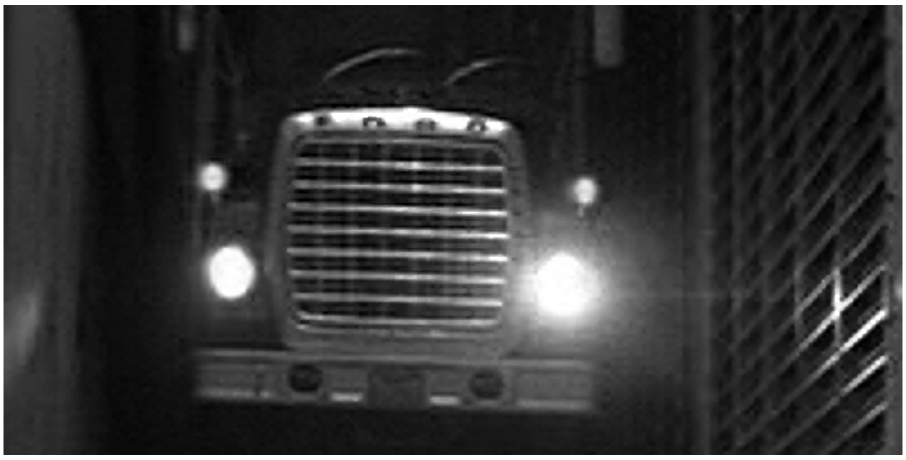

(c) 8x8 S-VSR

Figure A.14: 8x8 VSR on `Truck`. 200% of Figures 4.12(c)-(e). Due to the very large size of these frames (800×1600) an additional zoom of 125-150% might give a slightly better view.

Bibliography

[1] G. Aubert and P. Kornprobst, *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*, 2nd ed., ser. Applied Mathematical Sciences. Springer-Verlag, 2006, vol. 147.

[2] S. Baker and T. Kanade, "Limits on Super-Resolution and How to Break Them." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1167–1183, 2002.

[3] D. F. Barbe, "Charge-Coupled Devices," in *Topics in Applied Physics*, D. F. Barbe, Ed. Springer, 1980.

[4] E. Bellers and G. de Haan, *De-interlacing. A Key Technology for Scan Rate Conversion.* Elsevier Sciences Publishers, Amsterdam, 2000.

[5] M. Biswas, S. Kumar, and T. Nguyen, "Performance Analysis of Motion-Compensated De-Interlacing Systems," *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2596–2609, 2006.

[6] M. Biswas and T. Nguyen, "A Novel De-Interlacing Technique Based on Phase Plane Correlation Motion Estimation," *International Symposium on Circuits and Systems, ISCAS*, vol. 2, pp. 604–607, 2003.

[7] D. Bordwell and K. Thompson, *Film History: An Introduction.* McGraw-Hill, 1994.

[8] S. Borman and R. Stevenson, "Spatial Resolution Enhancement of Low-Resolution Image Sequences: A Comprehensive Review with Directions for Future Research," Laboratory for Image and Sequence Analysis (LISA), University of Notre Dame, Tech. Rep., July 1998.

[9] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High Accuracy Optical Flow Estimation Based on a Theory for Warping," in *Proceedings of the 8th European Conference on Computer Vision*, T. Pajdla and J. Matas, Eds., vol. 4. Prague, Czech Republic: Springer–Verlag, 2004, pp. 25–36.

[10] T. Brox, J. Weickert, B. Burgeth, and P. Mrázek, "Nonlinear structure tensors," *Image and Vision Computing*, vol. 24, no. 1, pp. 41–55, Jan. 2006.

[11] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr, "Variational Optic Flow Computation in Real-Time," *IEEE Trans. on Image Processing*, vol. 14, no. 5, pp. 608–615, 2005.

[12] 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.

[13] A. Bruhn, "Variational Optic Flow Computation: Accurate Modelling and Efficient Numerics," Ph.D. dissertation, Department of Mathematics and Computer Science, Saarland University, July 2006.

[14] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnörr, "Discontinuity-Preserving Computation of Variational Optic Flow in Real-Time." in *Scale Space and PDE Methods in Computer Vision, 5th International Conference, Scale-Space 2005, Proceedings*, ser. LNCS, R. Kimmel, N. A. Sochen, and J. Weickert, Eds., vol. 3459. Berlin: Springer, 2005, pp. 279–290.

[15] D. C. Burr and M. J. Morgan, "Motion deblurring in human vision," *Proceedings of the Royal Society B: Biological Sciences*, vol. 264, no. 1380, pp. 431–436, 1997.

[16] L. Capodiffero, "Interlaced to Progressive Conversion by Median Filtering," in *Signal Processing of HDTV, II*, L. Chiariglione, Ed. Elsevier Sciences Publishers, Amsterdam, 1990, pp. 677–684.

[17] M. Chahine and J. Konrad, "Motion-compensated interpolation using trajectories with acceleration," in *Proc. IS&T/SPIE Symp. Electronic Imaging Science and Technology, Digital Video Compression: Algorithms and Technologies 1995*, vol. 2419, 1995, pp. 152–163.

[18] T. Chan and J. Shen, "Mathematical models for local nontexture inpainting," *SIAM journal of appl. Math*, vol. 62, no. 3, pp. 1019–1043, 2002.

[19] S. Chaudhuri, Ed., *Super-Resolution Imaging*, ser. The International Series in Engineering and Computer Science. Springer, 2001.

[20] H.-F. Chen, S.-H. Lee, O.-J. Kwon, S.-S. Kim, J.-H. Sung, and Y.-J. Park, "Smooth Frame Insertion Method for Motion-Blur Reduction in LCDs," in *Multimedia Signal Processing, 2005 IEEE 7th Workshop on.* ieeexplore.ieee.org, Oct. 2005, pp. 581–584.

[21] V. Cheung, B. J. Frey, and N. Jojic, "Video epitomes," in *Proceedings of Computer Vision and Pattern Recognition, 2005. CVPR 2005.* ieeexplore.ieee.org, June 2005, pp. 42–49.

[22] K. J. Ciuffreda, A. Selenow, B. Wang, B. Vasudevan, G. Zikos, and S. R. Ali, ""Bothersome blur": A functional unit of blur perception." *Vision Research*, vol. 46, no. 6-7, pp. 895–901, 2006, e-published Dec. 6 2005.

[23] J. Cocquerez, L. Chanas, and J. Blanc-Talon, "Simultaneous Inpainting and Motion Estimation of Highly Degraded Video-Sequences," in *Scandinavian Conference on Image Analysis.* Springer-Verlag, 2003, pp. 523–530, lNCS, 2749.

[24] K. H. Cornog, G. A. Dickie, P. J. Fasciano, and R. M. Fayan, "Interpolation of a Sequence of Images Using Motion Analysis," U. S. Patent 6,665,450, 12 16, 2003.

[25] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Wiley Series in Telecommunications, 1991.

[26] A. Criminisi, P. Pérez, and K. Toyama, “Object Removal by Exemplar-Based Inpainting,” in *Conf. Computer Vision and Pattern Recog, CVPR'03*, vol. 2, Madison, WI, June 2003, pp. 721–728.

[27] G. Dane and T. Q. Nguyen, “Optimal Temporal Interpolation Filter for Motion-Compensated Frame Rate Up Conversion,” *Image Processing, IEEE Transactions on*, vol. 15, no. 4, pp. 978–991, 2006.

[28] G. de Haan, “IC for Motion-Compensated De-Interlacing, Noise Reduction, and Picture-Rate Conversion,” *IEEE Transactions on Consumer Electronics*, pp. 617–624, 1999.

[29] G. de Haan, P. W. A. C. Biezen, and O. A. Ojo, “An Evolutionary Architecture for Motion-Compensated 100 Hz Television,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 3, pp. 207–217, June 1995.

[30] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, “Dynamic Textures,” *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, February 2003.

[31] T. Doyle and M. Looymans, “Progressive Scan Conversion Using Edge Information,” in *Signal Processing of HDTV, II*, L. Chiariglione, Ed. Elsevier Sciences Publishers, Amsterdam, 1990, pp. 711–721.

[32] H. Dreier, “Line Flicker Reduction by Adaptive Signal Processing,” in *Signal Processing of HDTV, II*, L. Chiariglione, Ed. Elsevier Sciences Publishers, Amsterdam, 1990, pp. 695–701.

[33] E. Dubois and J. Konrad, “Estimation of 2-D Motion Fields from Image Sequences with Application to Motion-Compensated Processing,” in *Motion Analysis and Image Sequence Processing*, M. Sezan and R. Lagendijk, Eds. Kluwer Academic Publishers, 1993, pp. 53–87. [Online]. Available: citeseer.ist.psu.edu/112523.html

[34] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, 2nd Edition.* New York, NY: John Wiley and Sons Inc., 2001.

[35] J. Ehrhardt, D. Säring, and H. Handels, “Optical flow based interpolation of temporal image sequences,” pp. 61 442K1–8.

[36] Y. C. Faroudja, D. Xu, and P. Swartz, “Detector for Detecting Videosignals Originating from Motion Picture Film Sources,” European Patent WO 95/300 006 (EP 0 654 197), 12 22, 1994.

[37] S. Farsiu, M. Elad, and P. Milanfar, “Multiframe Demosaicing and Super-Resolution of Color Images,” *IEEE Transactions on Image Processing*, vol. 15, no. 1, pp. 141–159, 2006.

[38] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, “Fast and Robust Multiframe Super Resolution,” *IEEE Trans. on Image Processing*, vol. 13, no. 10, pp. 1327–1344, 2004.

[39] A. W. Fitzgibbon, “Stochastic Rigidity: Image Registration for Nowhere-Static Scenes,” in *Proc. of International Conference on Computer Vision*, 2001, pp. 662–670.

[40] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, “Learning Low-Level Vision,” *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000.

[41] B. Fuglede and F. Topsøe, “Jensen-Shannon Divergence and Hilbert space embedding,” in *Proceedings of the International Symposium on Information Theory*, Chicago, 2004, p. 31.

[42] S. Gallot, D. Hulin, and J. Lafontaine, *Riemannian Geometry*. Springer-Verlag, 1990.

[43] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Prentice Hall, 2002.

[44] L. D. Griffin and M. Lillholm, “Hypotheses for Image Features, Icons and Textons,” *International Journal of Computer Vision*, vol. 70, no. 3, pp. 213–230, Dec. 2006.

[45] P. Haavisto, J. Juhola, and Y. Neuvo, “Scan Rate Up-Conversion Using Adaptive Weighted Median Filtering,” in *Signal Processing of HDTV, II*, L. Chiariglione, Ed. Elsevier Sciences Publishers, Amsterdam, 1990, pp. 703–710.

[46] R. C. Hardie, K. J. Barnard, and E. E. Armstrong, “Joint MAP registration and high-resolution image estimation using asequence of undersampled images,” *Image Processing, IEEE Transactions on*, vol. 6, no. 12, pp. 1621–1633, Dec. 1997.

[47] H. He and P. Kondi, “An Image Super-Resolution Algorithm for Different Error Levels Per Frame,” *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 592–603, 2006.

[48] B. Horn and B. Schunck, “Determining Optical Flow,” *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[49] M. Irani and S. Peleg, “Motion Analysis for Image Enhancement: Resolution, Occlusion, and Transparency,” *Journal on Visual Communications and Image Representation*, vol. 4, no. 4, pp. 324–335, 1993.

[50] ITU, “ITU-R Recommendation BT.500-11: Methodology for the subjective assessment of the quality of television pictures,” Geneve, Switzerland, 6 2002.

[51] F. Kanters, “Towards Object-based Image Editing,” Ph.D. dissertation, Eindhoven Technical University, 2006.

[52] H. A. Karim, M. Bister, and M. U. Siddiqi, “Multiresolution Motion Estimation for Low-Rate Video Frame Interpolation,” *EURASIP Journal on Applied Signal Processing*, no. 11, pp. 1708–1720, 2004.

[53] S. Keller, F. Lauze, and M. Nielsen, "Variational Motion Compensated Deinterlacing," in *Proceedings of Statistical Methods in Multi-Image and Video Processing, ECCV 2006 Workshop*, 2006, pp. 49–56.

[54] S. H. Keller, F. Lauze, and M. Nielsen, "Motion Compensated Video Super Resolution," in *Scale Space and Variational Methods in Computer Vision, SSVM 2007 Proceedings*, ser. LNCS, F. Sgallari, A. Murli, and N. Paragios, Eds., vol. 4485. Berlin: Springer, 2007, pp. 801–812.

[55] S. Keller, F. Lauze, and M. Nielsen, "A Total Variation Motion Adaptive Deinterlacing Scheme," in *Scale Space and PDE Methods in Computer Vision, 5th International Conference, Scale-Space 2005, Proceedings*, ser. LNCS, R. Kimmel, N. Sochen, and J. Weickert, Eds., vol. 3459. Berlin: Springer, 2005, pp. 408–419.

[56] S. H. Keller, K. S. Pedersen, and F. Lauze, "Detecting Interlaced or Progressive Source of Video," in *Proceedings of the 2005 IEEE Seventh Workshop on Multimedia Signal Processing (MMSP)*, X. Zhuang, J. Sørensen, Q. Wu, Y.-Q. Shi, J. Ostermann, H. Man, and D. Goldgof, Eds. ieeexplore.ieee.org, 2005, pp. 181–184.

[57] S. H. Keller, "PDE Based Deinterlacing," Master's thesis, IT University of Copenhagen, Denmark, 2004, in Danish.

[58] ——. (2007) Selected electronic results of TSR experiments. [Online]. Available: http://image.diku.dk/sunebio/TSR/TSR.zip

[59] ——. (2007) Selected electronic results of VSR experiments. [Online]. Available: http://image.diku.dk/sunebio/VSR/VSR.zip

[60] S. Kim, N. Bose, and H. Valenzuela, "Recursive Reconstruction of High Resolution Image From Noisy Undersampled Multiframes," *IEEE Transactions on Acoustics, Speech, Signal Processing*, vol. 38, no. 6, pp. 1013–1027, 1990.

[61] J. J. Koenderink and A. J. van Doorn, "Representation of Local Geometry in the Visual System," *Biol. Cybernetics*, vol. 55, pp. 367–375, 1987.

[62] T. Kohlberger, C. Schnörr, A. Bruhn, and J. Weickert, "Domain Decomposition for Variational Optical Flow Computation," *IEEE Trans. Image Proc.*, vol. 14, no. 8, pp. 1125–1137, 2005.

[63] J. Kovacevic, R. Safranek, and E. Yeh, "Deinterlacing by Successive Approximations," *IEEE Transactions on Image Processing*, vol. 6, no. 2, pp. 339–344, 1997.

[64] F. Lauze, "Computational Methods For Motion Recovery, Motion Compensated Inpainting and Applications," Ph.D. dissertation, IT University of Copenhagen, 2004.

[65] F. Lauze and M. Nielsen, "A Variational Algorithm for Motion Compensated Inpainting," in *British Machine Vision Conference*, S. B. A. Hoppe and T. Ellis, Eds., vol. 2. BMVA, 2004, pp. 777–787.

[66] F. Lauze, S. Keller, and M. Nielsen, “A Method Of Adding Information To A Frame Or A Field In A Video Sequence,” U. S. Patent 11/239 697, 09 30, 2005, application.

[67] ——, “A Method Of Adding Information To A Frame Or A Field In A Video Sequence,” International PCT Patent PCT/EP2005/054 957, 09 30, 2005, application.

[68] M. Lillholm, M. Nielsen, and L. Griffin, “Feature-Based Image Analysis,” *International Journal of Computer Vision*, vol. 52, no. 2/3, pp. 73–95, 2003.

[69] Z. Lin and H.-Y. Shum, “Fundamental Limits of Reconstruction-Based Superresolution Algorithms under Local Translation,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 83–97, 2004.

[70] G. Lu, *Communication and Computing for Distributed Multimedia Systems.* Boston, MA: Artech House, 1996.

[71] B. Lucas and T. Kanade, “An Iterative Image Registration Technique with an Application to Stereo Vision,” in *International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[72] T. C. Lyon and J. J. Campbell, “Motion sequence pattern detector for video,” U. S. Patent 4,982,280, 1 1, 1991.

[73] M. W. Matlin and H. J. Foley, *Sensation and Perception*, 4th ed. Allyn and Bacon, 1997.

[74] M. J. Morgan and S. Benton, “Motion-deblurring in human vision,” *Nature*, vol. 340, pp. 385–386, 1989.

[75] D. Mumford, “Bayesian Rationale For The Variational Formulation,” in *Geometry-Driven Diffusion In Computer Vision*, B. M. ter Haar Romeny, Ed. Kluwer Academic Publishers, 1994, pp. 135–146.

[76] D. Munsil and B. Florian, “DVD Benchmark, Part 5 - Progressive Scan DVD,” Internet Article, 2005. [Online]. Available: http://www.hometheaterhifi.com

[77] M. Nadenau, S. Winkler, D. Alleysson, and M. Kunt. (2000) Human Vision Models for Perceptually Optimized Image Processing – A Review. [Online]. Available: http://citeseer.ist.psu.edu/nadenau00human.html

[78] O. A. Ojo and G. de Haan, “Robust motion-compensated video upconversion,” *IEEE Transactions on Consumer Electronics*, vol. 43, no. 4, pp. 1045–1055, Nov. 1997.

[79] A. J. Patti, M. I. Sezan, and A. M. Tekalp, “Robust methods for high-quality stills from interlaced video in thepresence of dominant motion,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 7, no. 2, pp. 328–342, April 1997.

[80] P. Perona and J. Malik, "Scale-Space and Edge Detection Using Anisotropic Diffusion," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, July 1990.

[81] S. Pigeon and P. Guillotel, "Advantages and Drawbacks of Interlaced and Progressive Scanning Formats," *CEC RACE/HAMLET*, 1995, deliverable no R2110/WP2/DS/R/004/b1.

[82] C. Poynton, *Digital Video and HDTV: Algorithms and Interfaces.* San Francisco, CA: Morgan Kaufmann/Elsevier, 2003.

[83] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 1996.

[84] J. Puttenstein, I. Heynderickx, and G. de Haan, "Evaluation of Objective Quality Measures for Noise Reduction in TV-Systems," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 109–119, February 2004.

[85] P. Robert, "Method for the temporal interpolation of images and device for implementing this method," U. S. Patent 5 214 751, 05 25, 1993.

[86] A. Roussos and P. Maragos, "Vector-Valued Image Interpolation by an Anisotropic Diffusion-Projection PDE," in *Scale Space and Variational Methods in Computer Vision, SSVM 2007 Proceedings*, ser. LNCS, F. Sgallari, A. Murli, and N. Paragios, Eds., vol. 4485. Berlin: Springer, 2007, pp. 104–115.

[87] M. Rucci, R. Iovin, M. Poletti, and F. Santini, "Miniature eye movements enhance fine spatial detail," *Nature*, vol. 447, no. 7146, pp. 851–854, June 2007. [Online]. Available: http://www.nature.com/nature/journal/v447/n7146/pdf/nature05866.pdf

[88] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259–268, 1992.

[89] G. Sapiro, *Geometric Partial Differential Equations and Image Analysis.* Cambridge University Press, 2001.

[90] R. Schultz and R. Stevenson, "Extraction of High Resolution Frames from Video Sequences," *IEEE Trans. on Image Processing*, vol. 5, no. 6, pp. 996–1011, 1996.

[91] R. J. Schutten and G. de Haan, "Real-Time 2-3 Pull-Down Elimination Applying Motion Estimation/Compensation in a Programmable Device," *IEEE Transactions on Consumer Electronics*, vol. 44, no. 3, pp. 930–938, 1998.

[92] E. Shechtman, Y. Caspi, and M. Irani, "Space-Time Super-Resolution," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 4, pp. 531–545, 2005.

[93] H. Shen, L. Zhang, B. Huang, and P. Li, "A MAP Approach for Joint Motion Estimation, Segmentation, and Super Resolution," *Image Processing, IEEE Transactions on*, vol. 16, no. 2, pp. 479–490, Feb. 2007.

[94] A. Skarabot, G. Ramponi, and L. Buriola, “FPGA architecture for a videowall image processor,” in *Proceedings of the SPIE International Symposium on Electronic Imaging*, vol. 4304, 2001, pp. 75–84.

[95] A. Skarabot, G. Ramponi, and D. Toffoli, “Image sequence processing for Videowall visualization,” in *Proceedings of the International Society for Optical Engineering*, vol. 3961, 2000, pp. 138–147. [Online]. Available: citeseer.ist.psu.edu/skarabot00image.html

[96] A. Srivastava, A. B. Lee, E. P. Simoncelli, and S.-C. Zhu, “On Advances in Statistical Modeling of Natural Images,” *Journal of Mathematical Imaging and Vision*, vol. 18, no. 1, pp. 17–33, Jan. 2003.

[97] J. Taylor, M. R. Johnson, and C. G. Crawford, *DVD Demystified*, 3rd ed. New York, NY: McGraw-Hill, 2006.

[98] A. Tekalp, *Digital Video Processing*. Prentice-Hall, 1995.

[99] G. Thomas, “A Comparison of Motion-Compensated Interlace-to-Progressive Conversion Methods,” *Signal Processing, Image Commun.*, vol. 12, no. 3, pp. 209–229, 1998.

[100] M. K. Thomsen, “Eksperterne, måleinstumenterne og de naive,” *Ingeniøren*, vol. 17, no. 3, pp. 2–3, April 2007, in Danish.

[101] R. Tsai and T. Huang, “Multiframe Image Restoration and Registration,” in *Advances in Computer Vision and Image Processing*, vol. 1, 1984, pp. 317–319.

[102] D. Tschumperlé and B. Besserer, “High Quality Deinterlacing using Inpainting and Shutter-Model Directed Temporal Interpolation,” in *Proc. of ICCVG International Conference on Computer Vision and Graphics*. Kluwer, 2004, pp. 301–307.

[103] D. Tschumperlé and R. Deriche, “Vector-Valued Image Regularization with PDEs: A Common Framework for Different Applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 4, pp. 506 – 517, 2005.

[104] A. Vadivel, M. Mohan, S. Surel, and A. K. Majumdar, “Object Level Frame Camparison for Video Shot Detection,” in *Proceedings of IEEE Workshop on Motion and Video Computing, Motion 2005*, vol. 2, 2005, pp. 235–240.

[105] S. Vedula, S. Baker, and T. Kanade, “Image-Based Spatio-Temporal Modeling and View Interpolation of Dynamic Events,” *ACM Transactions on Graphics*, vol. 24, no. 2, pp. 240 – 261, April 2005.

[106] VQEG. (2000, 3) Final report of the Video Quality Experts Group on the validation of objective models of video quality assessment. [Online]. Available: ftp://ftp.its.bldrdoc.gov/dist/ituvidq/old2/Final_Report_April00.doc

[107] Y. Wang, J. Ostermann, and Y. Zhang, *Video Processing and Communications*. Prentice-Hall, 2002.

[108] J. Whitaker, *DTV: The Revolution in Electronic Imaging.* New York: McGraw-Hill, 1998.

[109] M. Zhao and G. de Haan, "Subjective evaluation of de-interlacing techniques," in *Proceedings of SPIE – Image and Video Communications and Processing 2005*, A. Said and J. G. Apostolopoulos, Eds., vol. 5685, March 2005, pp. 683–691.

[110] S. C. Zhu and D. Mumford, "Prior Learning and Gibbs Reaction-Diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 11, pp. 1236 – 1250, Nov. 1997.