Peter Innerhofer

A Bilevel Sparse Coding Approach for Super Resolution

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's Degree program

Computer Science

submitted to

Graz University of Technology

Thesis supervisor

Univ.-Prof. Dipl.-Ing. Dr. Thomas Pock

Graz, Austria, October 2014

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am …………………………… ………………………………………………..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………..
date (signature)

Abstract

Single image super resolution is a fundamental research topic and refers to the process of up-sampling or upscaling of a single raster graphics image. The developed methods start from simple interpolation-based filtering, inverse problem statement and example-based methods to systems utilizing sparse representation. In this work we further develop the Super Resolution (SR) method utilizing sparse representation by incorporating it in a bilevel program. Formulating the SR problem via sparse representation as bilevel programing problem has various advantages over the initially defined joint sparse coding scheme by Yang et al.[YWHM10]. The joint sparse coding scheme trains two dictionaries, a low-resolution and a high resolution dictionary in the concatenated feature space in a single instance leading to a suboptimal sparse decomposition in the test case where only the low-resolution feature space is given. In contrast our bilevel program learns the two dictionaries such that they are optimal in both feature spaces individually. In the test case sparse decomposition in the low resolution feature spaces is therefore optimal and we can show significant improvements over the joint sparse coding scheme developed by Yang et al. Additionally our bilevel training scheme implicitly learns the mapping function from low to high-resolution feature space without an explicit definition or inversion of a forward model. This is advantageous since this mapping function is non-linear. We show that our bilevel program can compete with state-of-the-art algorithms.

Keywords. Single Image Super Resolution, Sparse Coding, Sparse Representation, Sparse Decomposition, Bilevel Optimization, Bilevel Program

Kurzfassung

Super Resolution gehört zur Grundlagenforschung im Bereich der Bildrekonstruktion und beschreibt das Vergrößern von einzelnen Rastergrafiken/natürlichen Bildern. Die herangezogenen Methoden reichen von einfacher interpolativer Filterung über inverse Problemdefinition und beispielbasierten Ansätzen hin zu Systemen, die Sparse Approximation einsetzen. In dieser Arbeit entwickeln wir den Ansatz der Sparsen Approximation weiter, indem wir es in ein Bilevel Optimierungs- progamm einbetten. Die wegweisende Arbeit von Yang et al.[YWHM10] zeigt die Stärken der Methode der Sparsen Approximation angewendet auf das Gebiet der Bildvergrößerung auf. Dabei beinhaltet ihr Ansatz eine grundlegende Schwäche. Sie verwenden ein suboptimales kombiniertes Training, wobei zwei Wörterbücher erstellt werden, eines für den hochaufgelösten und eines für niedrigaufgelösten Bildraum. Durch ihr kombiniertes Training sind die Wörterbücher aber nicht optimal in den einzelnen Bildräumen was zum Nachteil beim Test der Vergrößerung führt, da hier nur das niedrigaufgelöste Bild vorhanden ist. Unser zweischichtiges mathematisches Optimierungsprogramm hingegen lernt die Wörterbücher so, dass sie in beiden Bildräumen optimal sind. Der Testfall, in dem nur das niedrigaufgelöste Bild vorhanden ist, ist damit mathematisch optimal und wir können signifikante Verbesserungen zum ursprünglichen Ansatz von Yang et al. präsentieren. Zusätzlich lernt unser zweischichtiges Optimierungsprogramm die Transformation vom niedrigaufgelösten zum hochaufgelösten Bildraum ohne diese explizit zu definieren. Der Vorteil dabei, diese Transformation ist schwer zu modellieren und nicht linear. Abschließend zeigen wir, dass unsere Ergebnisse auf Augenhöhe mit den modernsten Methoden ist.

Schlagwörter. Sparse Coding, Super Resolution, Bilevel Optimization

Acknowledgments

This thesis has been made possible thanks to many generous people which i am grateful and acknowledge them here.

Special thanks goes to my mentor, teacher and supervisor Dr. Thomas Pock for his thoughtful comments and guidance. He was always available for valuable advise, outstanding support and relentless encouragement.

Further, I want to give credits to the institute for computer graphic and vision which gave me a nice workplace, a powerful workstations, financial help and technical support.

Finally this thesis would not have been possible without the most devoted, supportive and caring family. To them I send my deepest gratitude, appreciation for their sustained encouragement and my endless love.

Contents

List of Symbols

A	System Matrix, product of a Sub-sampling, a blurring and alignment matrix
D_h	The high-resolution Dictionary
D_l	The low-resolution Dictionary
D	Synthesis Dictionary
L	Lipschitz constant
P_G	Global Projection Matrix
S, B, W	Sub-sampling, blurring and alignment matrices
Y_k	Nearest Neighbors concatenated in a matrix
$\Phi(x)$	Regularization function of signal x
α, β	Scalar convexity parameter, used by Inertial Proximal Algorithm for strongly convex Optimization (IPIASCO)
ϵ	Scalar parameter of the $l_{1,e}$-norm
η	Learning rate
λ	Scalar regularization parameter
$\mathcal{T}_\lambda$	Scaled soft-threshold shrinkage operator
∇	Nabla operator
$\otimes$	Kronecker product
∂	Sub-differential
$\boldsymbol{\alpha}$	Sparse representation vector
$\boldsymbol{f^h}$	High-Resolution features vector (zero-mean patches), input to the bilevel program
$\boldsymbol{f^l}$	Low-Resolution features vector, input to the bilevel program
$\boldsymbol{n}$	Noise vector, typically Gaussian i.i.d.
$\boldsymbol{w}$	Reconstruction weights vector
$\boldsymbol{x}$	Signal - a column vector over the real numbers
$\boldsymbol{y}$	Degraded or observed signal - a column vector
f, f'	Function and its derivative
m, n, q, p	Size of matrices - scalar variables

List of Figures

List of Tables

List of Algorithms

1. Introduction

Contents

1.1. Motivation

Many digital image applications demand High Resolution (HR) images or videos as an input for signal processing and analysis or for human interpretation. Although there exist several classes of resolution in the context of digital images such as spatial, spectral or temporal resolution, here we focus on spatial resolution. A digital image is made up of small picture elements called pixels and spatial resolution refers to the density of pixels per unit area. Higher pixel density mostly signifies more image information, possibly higher frequencies and structural information. Such HR images can be obtained by high quality video acquisition systems. These systems are limited by their components, mainly the image sensor and the optical system but these components can be very expensive. Images from less costly sources like the Internet, smart phones, surveillance, medical images, satellites or old content (PAL/NTSC) often do not have the resolution needed for adequate processing, analysis, zooming or displaying capacity. In this cases SR can play an important role as it can improve the resolution of such content[YH11]. Additionally SR is a fundamental research topic comparable with image deblurring, inpainting, denoising or image restoration in general as these subjects can give proof-of-concept for recent scientific developments.

1.2. Super Resolution

Originally, SR refers to the process of upscaling (or up-sampling) a digital video. The basic idea in SR is to combine the non-redundant information in the Low Resolution (LR) frames and form a HR image. Figure 1.1 shows a simplified sketch how a basic SR reconstruction algorithm works. In 1984 Tsai and Huang [HT84] presented the first super resolution reconstruction of an image sequence by aligning the degraded LR image frames and merging them in the frequency domain to form the HR image sequence. The term “Super Resolution” was first mentioned by Irani et al. in 1991 in their work “Improving resolution by image registration”[IP91]. Historically, super resolution was mainly applied on multi-frame images (videos) and hence referred to, as classical super resolution. Later research moved on to the more challenging up-sampling of single images. In the literature, single image super resolution is also referred to as image interpolation or image hallucination. Task-driven SR algorithms were developed for specific problems in areas such as surveillance, where inspection and recognition of face images or license plates is required. These problems can be better constrained and special image priors can be exploited. In general, SR is a computational complex and numerically ill-posed problem. This is even more true for single image SR since there is no additional information except the image itself. In this work we focus on single image super resolution for natural images.

A main concern in single image SR is to find an image prior to constrain the problem. Systems like [FREM04],[AD05] and [UPWB10] try to exploit natural image priors based on intuitive understanding of natural images as they consist mainly of flat regions separated by sharp edges[BM87]. Others focus on statistical analysis and distribution of edges to regularize the problem[Fat07][SSXS08]. Systems incorporating example-based image priors like [FJP02],[CYX04] and [BRGA12] also have shown great success. Since example-based systems require large training sets in storage, single image SR systems utilizing sparse representations [YWHM10] [ZEP12][TDG13] have become attractive as they reduce the stored data significantly. Very recently, SR systems modeling the entire SR-pipeline by neuronal networks have shown yet more superior results[KH12][DLHT14].

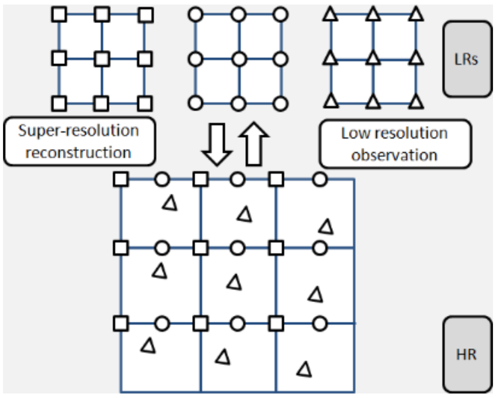


Figure 1.1.: The basic idea in SR reconstruction is to combine the non-redundant information in the LR frames by employing the sub-pixel shift between single video frames (image adopted from [YH11]).

1.3. Contribution

Single image Super Resolution is an active field of research and lately SR algorithms based on sparse coding have become state-of-the-art methods[YH11]. Sparse Representation or Sparse Coding (SC) was originally developed for Compressed Sensing/Compressive Sampling[FR11]. The main idea in Compressive Sampling is to perform compression directly while capturing data. It is a paradigm that tries to surpass Shannon's sampling theorem and create a new type of sampling theory[CW08]. With the aid of a leaned dictionary, SC can successfully recover a signal of length n with $k \ll n$ nonzero coefficients. In SC one learns an over-complete set of bases on the input signal such that the signal can be sparsely represented by these bases of the dictionary. This leads to a dimensionality reduction for the benefit of any signal transmission or compression algorithm. Such dictionaries can be used to tackle the Super Resolution problem, recent examples are [YWHM10][YWL+12b][HQZ13][ZEP12] or [TDG13]. The seminal work of Yang et al.[YWHM10] jointly learns two dictionaries, one for high-resolution patches and one for low-resolution patches in a concatenated feature space. In the test case, Yang et al. find the sparse representation on the LR image facilitating the LR dictionary and use the same representation to reconstruct the HR image utilizing the HR dictionary. As they mention in their work, through the joint leaning process, the dictionaries are only optimal in the concatenated feature space but not in each individual space. However, in the test case of an upscaling process, only the LR input is given and one can only find the sparse representation in the

LR feature space. Thus this learning scheme is suboptimal.

In comparison we propose a bilevel program for the dictionary learning following the works of Yang et al.[YWL+12b][YWL+12a]. Bilevel programming was originally developed in game theory and NP-hard problems like the traveler-salesman can be efficiently solved with such programming techniques[MSS04][Bar98]. A bilevel program is a hierarchical optimization problem that contains an optimization problem in the constraint of another optimization problem[BM73]. It consists of a upper-level objective function and a lower-level objective, both can have constraints added[VC94]. In our case we have two closely related dictionary learning problems but one goal, namely to reconstruct high-quality HR images. In this work we develop a bilevel program for learning a low- and a high-resolution dictionary coupled by a common sparse vector. This bilevel optimization formulation is designed to be optimal in both feature spaces individually which leads to better results in the reconstruction.

1.4. Outline

This work is organized as follows. First we give an overview of SR techniques and describe the leading works in this field. Furthermore, we give an introduction to optimization techniques used by our SR systems in chapter 2. Next we present two similar sparse coding approaches incorporated to a bilevel optimization formulation and we derive an algorithm for each. The first approach equips the lower-level objective by a smoothed $l_{1,\epsilon}$-regularization delineated in chapter 3, while the second approach follows the active set method detailed by Yang et al.[YWL+12b] later in the same chapter. In chapter 4 we compare the two algorithms with each other and with state-of-the-art methods. We present the main features of our implementation regarding the color treatment, the datasets we use and the image quality assessment. We conclude in chapter 5 and highlight further work.

2. Fundamentals

Contents

2.1. Notation

First we want to clarify some basic notations. Capital letters are used for matrices and matrix functions like $F(X), X, A, \ldots$ while lower-case letters are preserved for vector functions $f(X), f(\boldsymbol{x}), \ldots$, bold lower-case letters are used for vectors $\boldsymbol{x}, \boldsymbol{c}^T$ and non-formated letters signify scalars $\lambda, \ldots$.

2.2. Super Resolution Interpolation

In Super Resolution (SR) we operate on raster graphic images, sometimes referred to as bitmap or pixmap images. They consist of a rectangular grid of discrete pixel values with an associated bit depth, normally in the 8-bit range[Fol96]. In image interpolation, a pixel value is interpreted as a discrete data point of a continuous interpolation function. Basic image interpolation algorithms such as nearest-neighbor, bilinear or bicubic interpolation approximate the missing pixel information from their most proximate neighbors in the 2D pixel grid. Figure 2.1 shows three basic interpolation kernels and figure 2.2 shows the result of these basic interpolation algorithms applied on the “lena” image.

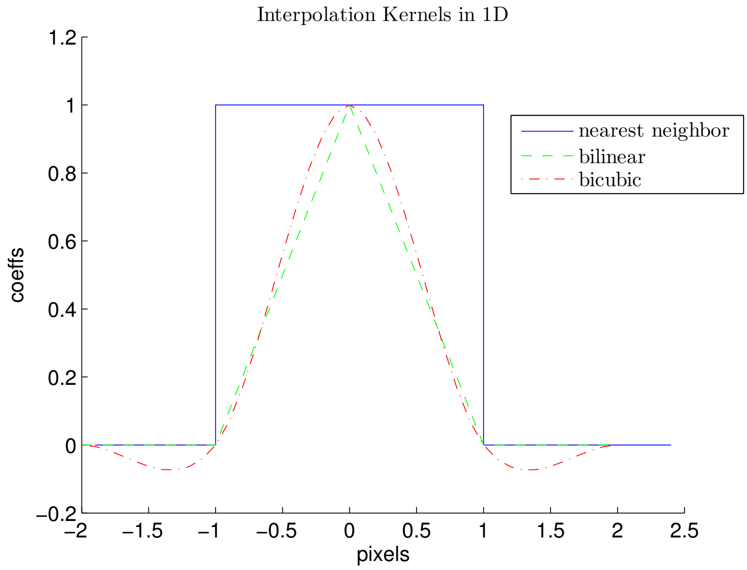
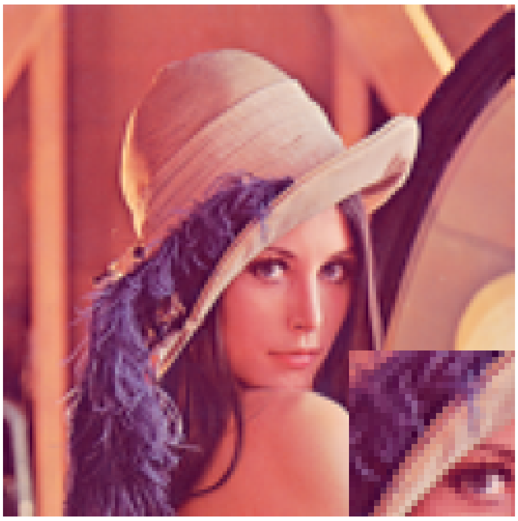


Figure 2.1.: Basic interpolation kernels in 1D. Blue is the nearest-neighbor, green the bilinear and red shows the bicubic interpolation kernel.

These basic image interpolation kernels are data-invariant linear filters with low complexity. They are unable to adapt to varying pixel structures and therefore suffer from blurring edges, textured regions or details in general. More advanced image interpolation algorithm such as the New Edge Directed Interpolation (NEDI)[LO00], Soft-decision Adaptive Interpolation (SAI)[ZW08] or the interpolation via Regularized Local Linear Regression (RLLR)[LZX+11] can partially overcome these limitations. The basic idea in NEDI for example is to use the local covariance coefficients computed on a Low Resolution (LR) patch to adapt the interpolation coefficients forming a High Resolution (HR) image pixel. This approach is capable of tuning the interpolation coefficients to match an arbitrary directed step edge. SAI in contrast estimates a group of pixels rather than a single pixel. This approach adapts to varying scene structures using a 2D piecewise auto-regression model where the model parameters are estimated at a moving

(a) original



(b) nearest

(c) bilinear

(d) bicubic

Figure 2.2.: Result of basic image interpolation algorithms upsampled by a magnification factor of 3. First image shows the original input file of size 254 x 254 pixels. The following images from left to right show interpolation results of nearest-neighbor (29.1dB), bilinear (30.1dB) and bicubic(31.4dB) interpolation respectively.

window in the LR input image. Additionally, the learned model is enforced by a soft-decision process applied on a block of pixels in the LR observation and on the HR estimate. Their approach preserves spatial coherence in the estimate and reduces common visual artifacts such as blurring and ringing. The ideas of SAI have been incorporated in other algorithms like the robust version RSAI[ZFW13].

The work of Dong et al.[DZLS13] incorporates the ideas of sparse coding in image interpolation. They uses a Principal Component Analysis (PCA) dictionary in addition to the known redundancies in natural images to estimate a high resolution image. Dong et al. incorporate a auto-regression model like SAI but extend it to non-local patches within the image. The sparse coding model and the non-local auto regression model are then combined in a complex optimization framework consisting of PCA dictionary learning, solving the auto regression model within a regularized least-squares formulation, sparse decomposition done with FISTA[BT09] followed by a conjugate gradient minimization. Their algorithm achieves good results but has a rather slow runtime due to the high computational complexity.

Image interpolation and super resolution are closely related. One could say that image interpolation is a subtask of super resolution by omitting image degradations such as blur and noise but separating these two fields of research becomes increasingly difficult. While in image interpolation the focus is set on the up-sampling process itself, super resolution aims to address all undesired effects of image degradation including resolution degradation, blur and noise. A SR algorithm typically models three parts, the up-sampling or interpolation, a deblurring and a denoising step. Image interpolation is still a highly active field of research and nowadays incorporates many machine learning techniques[SH12].

2.3. Super Resolution as an Inverse Problem

Super Resolution attempts to reconstruct a HR image from a LR observation. This type of a formulation is called an inverse problem. To solve an inverse problem in general, one requires the formulation of a forward model (or observation model). In the case of SR, the most common linear forward model is given by

$$\boldsymbol{y} = A\boldsymbol{x} + \boldsymbol{n}. \tag{2.1}$$

where $\boldsymbol{y}$ is the LR observation, A the system matrix, $\boldsymbol{x}$ the HR estimate and $\boldsymbol{n}$ the remaining noise. The system matrix A is the product of a sub-sampling matrix or down-sampling operator S, a blurring or anti-aliasing operator B and an optional alignment operator W for classical multi-image SR, hence $A = SBW$. The forward model 2.1 for SR is an underdetermined system and difficult to invert. Having defined a forward model one can formulate a cost function which ensures that the final solution is "close" to the measured observation. The cost function for (2.1) is given as

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} J(x) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - A\boldsymbol{x}\|_2^2, \tag{2.2}$$

where the noise $\boldsymbol{n}$ is modeled as additive zero-mean white Gaussian noise, and therefore the cost function is equipped with the quadratic norm. This cost function is called the reconstruction constraint and can be interpreted as the Maximum Likelihood (ML) estimator $p(\boldsymbol{x}|\boldsymbol{y})$[EF97] given the observation $p(\boldsymbol{y})$. An algorithm minimizing (2.2) must necessarily invert the linear forward model (2.1). This can be done by utilizing the pseudo inverse of the system matrix, hence $(A^T A)^{-1}$. Since A is underdetermined, $A^T A$ can be ill-conditioned and inverting it can be numerically unstable and amplify the noise in the singular vectors of $A^T A$. Since a robust SR algorithm is desired, adding a regularization to the cost function is a common way to stabilize the SR reconstruction,

$$J(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{y} - A\boldsymbol{x}\|_2^2 + \lambda\Phi(\boldsymbol{x}). \tag{2.3}$$

The regularization in (2.3) poses a constraint on the space of solutions of $\boldsymbol{x}$. From a Bayesian viewpoint this can be seen as an image prior $p(\boldsymbol{x})$ and therefore minimizing (2.3) can be interpreted as the Maximum A-posteriori Pobability (MAP) estimator. In literature common regularizations are Tikhonov regularizer $\Phi(\boldsymbol{x}) = \|T\boldsymbol{x}\|_2^2$, Total Variation (TV) regularizations $\Phi(\boldsymbol{x}) = \|\nabla\boldsymbol{x}\|_1$ and many more. This optimization problem can be solved by various algorithms including gradient decent methods like Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)[BT09] or interior-point methods like the primal-dual algorithm of Chambolle and Pock[CP11]. For a regularization equipped with the quadratic norm such as Tikhonov regularization, the problem can be solved explicitly and reduces to a ridged regression. SR application solving (2.3) are for example Farsiu et al. [FREM04], Mitzel et al. [MPSC09], Unger et al. [UPWB10] and Innerhofer et al.[IP13]

2.4. Super Resolution via Learning Based Regularization

Modeling SR as an inverse problem with a generic global regularization results in a fast and robust algorithm. The drawback of this rather basic approach is that it cannot infer novel image details lost in a down-sampling process. Especially for single image SR, the regularization becomes crucial and a local example based non-parametric image prior can outperform a generic global regularization, particularly for higher up-sampling factors. In leaning based SR one tries to find a non-parametric local image prior which can infer novel image details.

Backer and Kanade stated in there seminal work "Limits on super-resolution and how to break them"[BK02], that with increasing magnification factors the reconstruction constraint combined with a smoothness prior becomes less meaningful. The HR images of such a system result in very little high-frequency content. By using a "recognition prior" exploited by learning face images and by incorporating additional similar face images to the reconstruction constraint, Backer et al. could outperform former SR systems. They called their SR algorithm a hallucination algorithm.

The goal of learning based SR is to estimate HR details that are not present in the LR observation and can not become visible by simple sharpening. An early work in example-based SR is the system of Freeman et al.[FJP02] where they use example patches directly in the upscaling process. They generate a huge training set of low and high-resolution patch-pairs for every possible LR image patch. Each patch pair is connected via the observation model (2.1): $\boldsymbol{y_i} = A\boldsymbol{x_i} + \boldsymbol{n}$.

In inference, just taking the nearest LR patch from the training set and using the corresponding HR patch to form the HR estimate would lead to poor results with many disturbing artifacts. They add a probabilistic model to account for spatial coherence between overlapping HR patches. The probabilistic model proposed by Freeman et al. is a Markov Random Field (MRF). Figure 2.3 shows this MRF model where the $\boldsymbol{y_i}$-nodes are LR observed input patches, the HR estimated patches $\boldsymbol{x_i}$ are "hidden" nodes and lines indicate statistical dependencies between nodes. The optimal HR patch at each $\boldsymbol{x_i}$-node is the collection which maximizes the Markov's network probability. The exact solution to the MRF can be computationally intractable for which reason an approximate, iterative algorithm called

Belief Propagation (BP) [YFW00] was employed. BP is a message passing algorithm specialized on graphical models such as MRF or Bayesian Networks[Pea88].

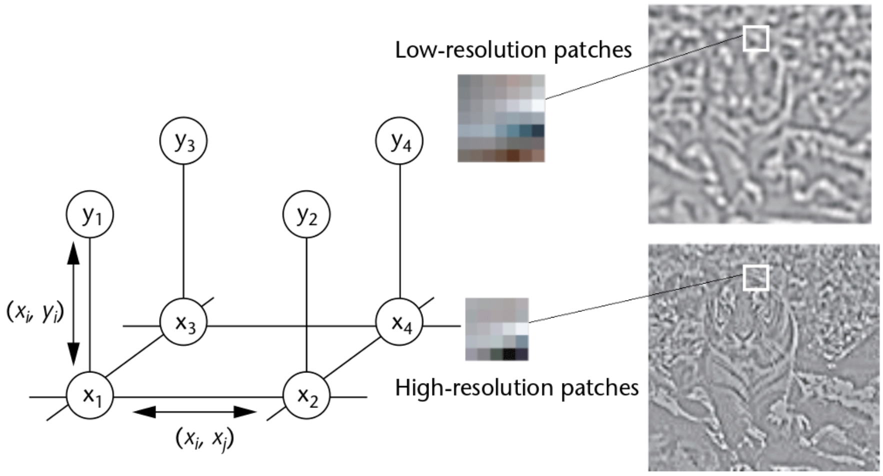


Figure 2.3.: MRF model used by Freeman et al.[FJP02]. The LR patches, located at nodes $\boldsymbol{y_i}$ are the observed inputs. The HR patches donated as "hidden" nodes $\boldsymbol{x_i}$ are the estimates. The lines indicate statistical dependencies between the nodes.

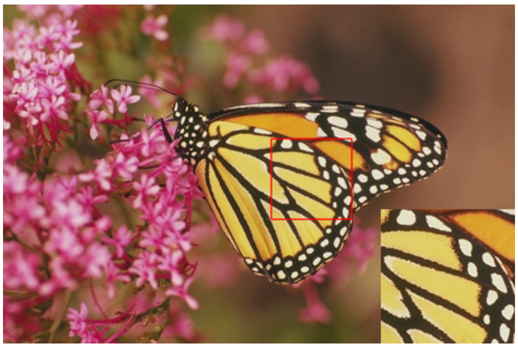
Another effective approach in example based SR is the method of Chang et al.[CYX04], called Neighborhood Embedding (NE) through Locally Linear Embedding (LLE). NE with LLE was originally developed in Manifold learning and uses local patches to reconstruct the input. Suppose we have a high-dimensional data space provided with sufficient data points, the local geometry of a new patch can be identified by the reconstruction weights of local or *similar* patches from the dataset. The reconstruction weight is a measurement matrix with which a data point is reconstructed from its Nearest Neighbors (NN) minimizing the reconstruction error. Equation (2.4) gives the formula to calculate the reconstruction weight $\boldsymbol{w_p}$ for a LR patch $\boldsymbol{y_p}$ utilizing the K-NNs concatenated in the matrix Y_k, donated as

$$\boldsymbol{w_p^*} = \arg\min_{\boldsymbol{w_p}} \|\boldsymbol{y_p} - Y_k \boldsymbol{w_p}\|_2^2. \tag{2.4}$$

This is a least squares problem on a linear system of equations and has a closed-form solution which leads to an efficient algorithm. The LLE used by [CYX04] roughly consists of two steps. First find K nearest neighbors in the LR feature space and calculate the reconstruction weights minimizing the reconstruction error, following equ. (2.4). Use the same reconstruction weights and the appropriate high-resolution K-NNs to compute the HR patches here referred as embeddings.

In [CYX04] these HR embeddings are then used to form the HR image and are averaged in overlapping regions.

A recent enhancements of a NE approach is the work of Bevilacqua et al. [BRGA12]. Their system is based on [CYX04] but in contrast they use Non-Negative Least Square (NNLS) rather than LLE. NNLS is similar to LLE but adds a non-negative inequality constraint to the least-square fitting of the reconstruction weights. Figure 2.4 gives an example result of these NE algorithms. It is interesting to see that in this example and for most of our test images, LLE outperforms the NNLS approach, but this could be due to the lack of parameter tuning since we used only the default settings.



(a) original

(b) bicubic (28.1dB)

(c) NE+LLE (29.5dB)

(d) NE+NNLS (29.6dB)

Figure 2.4.: Result of NE algorithms upsampled by a magnification factor of 3. First image shows the original input file of size 762 x 504 pixels. The following images from left to right show SR results of bicubic interpolation (28.1dB), NE with LLE (29.5dB) and NE with NNLS(29.6dB) approach, respectively.

2.5. Super Resolution via Sparse Representation

Sparse Representation or Sparse Coding (SC) is a method first developed in the field of Compressed Sensing[FR11]. The idea is to learn a set of over-complete bases called dictionary and use a linear combination of few of these bases to estimate a signal. Representing data in an over-complete dictionary is called sparse representation or SC and the bases or entries in the dictionary are called atoms. An over-complete dictionary is a redundant representation of data, meaning that we have more atoms than dimensions in the signal space and a signal can be represented by more than one combination of atoms. This promises to represent a wider range of signal phenomena than just using a complete set of bases[RBE10].

Sparse Coding has been successfully applied to various image reconstruction task including image denoising[EA06], inverse half-toning [MBP12], image deblurring[CDMBP11], restoration of missing pixels[AEB06] or artistic image transforms/conversions[WZLP12a].

The seminal work of Yang et al.[YWHM10] first applied SC to SR. They jointly learn a low- and high-resolution dictionary D_l, D_h from a large training set of patches. The patch pairs are connected via the observation model (2.1) and features are eventually taken from the LR patches. In the reconstruction one seeks a linear combination of LR atoms representing a LR patch or feature such that the number of dictionary atoms in use is small, avoiding overfitting. Thus, a sparsity inducing norm has to be included as a regularization. The sparse vector found by this scheme on the LR observation is then used to from the HR patch utilizing the HR dictionary. A convex relaxation to the sparse decomposition problem in the unconstrained formulation is given as

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \underbrace{\frac{1}{2}\|\boldsymbol{y} - D_l\boldsymbol{\alpha}\|_2^2}_{\text{data fitting term}} \quad + \quad \underbrace{\lambda\|\boldsymbol{\alpha}\|_1}_{\text{sparsity inducing term}} \tag{2.5}$$

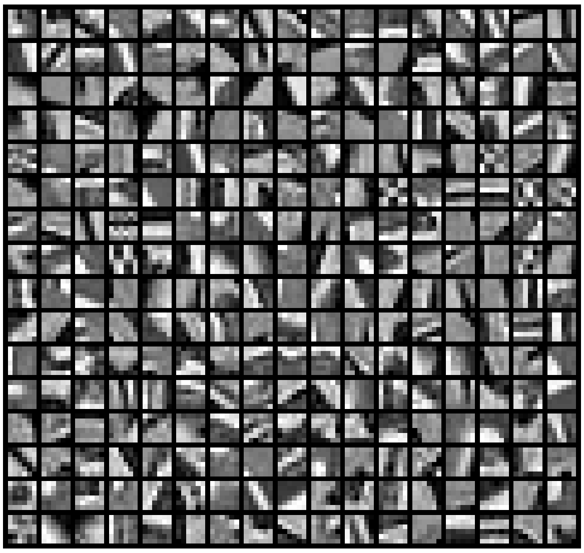
where $\boldsymbol{y}$ is the LR observation, D_l the LR dictionary, $\boldsymbol{\alpha}$ the sparse vector and λ a parameter controlling the sparsity penalty. At this point we note that D_l has a dimension of $m \times n$ where $m \ll n$ making the linear system under-determined. Therefore we have more atoms n than dimensions in the signal space m and the dictionary is said to be over-complete. The same is true for the HR dictionary. The HR patch $\boldsymbol{x}$ is than recovered using the HR dictionary D_h and the sparse vector $\boldsymbol{\alpha}$ found by the decomposition s.t. $\boldsymbol{x} = D_h\boldsymbol{\alpha}$. In literature equation (2.5) is known as the Least Absolute Shrinkage and Selection Operator (LASSO) problem and can be solved with various algorithms including Least-angle Regression (LARS)[EHJT04], FISTA[BT09], primal-dual [CP11] and many more[YGZ+10][BJMO12]. The process of estimating a sparse vector satisfying a linear system of equations is referred to as sparse approximation, sparse decomposition or dictionary inference. Equ. (2.5) consists of a sparsity inducing term to assure that the vector $\boldsymbol{\alpha}$ is sparse (most entries equal zero) and therefore the under-determined linear system of equation represented by the data fidelity term is solved by using only a few atoms of the dictionary fitting the input vector $\boldsymbol{y}$. The sparsity inducing term can vary depending on the problem statement. Common regularizations are the l_1-norm or the l_0-pseudo-norm but also the elastic-net regularization, mixed l_1/l_p-norms and group LASSO can be employed[BJMO12].

SR via sparse representation like [YWHM10] can be seen as a further development of the example based regularization. Example based SR systems like [FJP02] use image patches directly as priors and therefore require the large sets of patch pairs in storage. In SC the learned dictionaries form an over-complete set of bases and reduce the training result stored significantly. Moreover, due to the redundancy the dictionary is still flexible enough to account for most signal phenomena. Sparse representation can also be seen as a inference-by-synthesis model which does not need to solve an ill-conditioned inverse model but rather synthesizes a signal through a well-conditioned model. Figure 2.5 shows a dictionary learned on high resolution patches with 1024 atoms each with a size of 6×6 pixels.

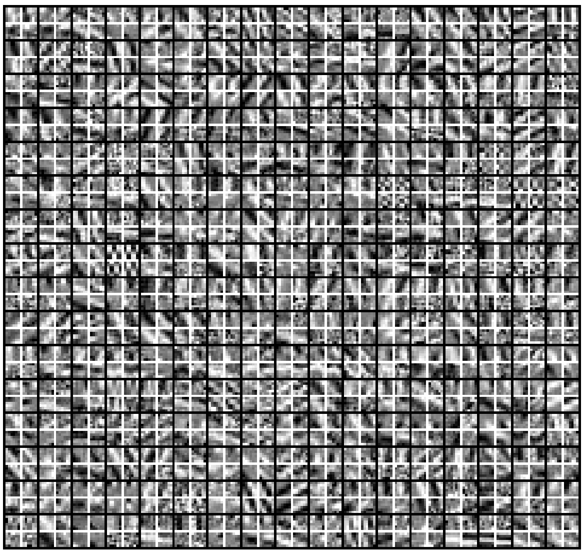
The difficult task in SC is the dictionary learning. The problem statement is NP-hard and no generic solver can be used. The dictionary learning problem in the unconstrained formulation in a single feature space is given as

$$\min_{D,\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{x} - D\alpha\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_p, \tag{2.6}$$

where p can ether be the l_0 pseudo-norm, the l_1-norm and various combinations of group l_1/l_p-norms. This optimization problem is non-convex and non-linear since it has to minimize both the dictionary D and sparse vector $\boldsymbol{\alpha}$ simultaneously. The standard solution is to split the subject into two separate convex sub-problems and alternatively optimize both. First, one initializes the dictionary with random



(a) HR dictionary



(b) LR dictionary

Figure 2.5.: Learned high- and low resolution dictionary, each with 1024 atoms of size of 6×6 pixels. We only show the first 272 atoms to give better details. The dictionaries where trained with our $l_{1,\epsilon}$-regularized bilevel program. The LR features consists of first- and second order central differences in horizontal and vertical direction.

sampled patches and solves the sparse decomposition problem optimizing (2.5) to find the optimal $\boldsymbol{\alpha}$. Subsequently one optimizes D while keeping $\boldsymbol{\alpha}$ fixed. The optimization in regard to the dictionary D while the sparse representation vector $\boldsymbol{\alpha}$ is fixed, is known as a Quadratically Constrained Quadratic Programming (QCQP)[YWL+12a]. Since the dictionary learning problem is non-convex and non-linear, one can only find a local minimum of $\boldsymbol{\alpha}$ and D[LBRN06]. Note that we have given the unconstrained dictionary learning problem. Usually a constraint on the dictionary atoms is added to prevent trivial solutions, hence $\|D(:,k)\|_2 \leq 1$, for $k = 1, 2, \ldots, K$. A trivial solution satisfying the dictionary learning problem 2.6 is for example the dictionary being the identity matrix $D = I$ while the sparse vector is the input $\alpha = x$.

2.5.1. Sparse Coding for Coupled Feature Spaces

In the case of SR we actually have two feature spaces, one high- and one low-resolution signal space, meaning $\mathcal{X}$ and $\mathcal{Y}$ respectively. The seminal work of Yang et al.[YWHM10] proposes to learn two dictionaries D_l, D_h for each feature space. These two spaces are tied by a mapping function $\mathcal{F}$. The simplest case is shown in the observation model (2.1). Their goal is to collaboratively learn coupled dictionaries (D_l, D_h) such that the sparse representation of the LR dictionary can be used to reconstruct the paired signal in the HR space. Yang et al. proposed a method which essentially concatenates the two feature spaces and transforms the dictionary learning problem in two separate feature spaces to a standard SC problem (2.6) in a single feature space. The following formula ensures that the common sparse representation $\boldsymbol{\alpha_i}$ reconstructs both the LR feature $\boldsymbol{y_i}$ and the HR patch $\boldsymbol{x_i}$,

$$\min_{D_h, D_l, \{\boldsymbol{\alpha_i}\}_{i=1}^N} \sum_{i=1}^{N} \frac{1}{2} \left(\|\boldsymbol{y_i} - D_l \boldsymbol{\alpha_i}\|_2^2 + \|\boldsymbol{x_i} - D_h \boldsymbol{\alpha_i}\|_2^2 \right) + \lambda \|\boldsymbol{\alpha_i}\|_1. \tag{2.7}$$

Grouping the two reconstruction errors of (2.7) leads to the standard SC scheme of (2.6) in the concatenated feature space of $\mathcal{X}$ and $\mathcal{Y}$, donating

$$\bar{\boldsymbol{x}}_{\boldsymbol{i}} = \begin{bmatrix} \boldsymbol{x_i} \\ \boldsymbol{y_i} \end{bmatrix}, \bar{D} = \begin{bmatrix} D_h \\ D_l \end{bmatrix}, \tag{2.8}$$

$$\min_{\bar{D},\{\boldsymbol{\alpha_i}\}_{i=1}^{N}} \sum_{i=1}^{N} \frac{1}{2}\|\bar{\boldsymbol{x}}_{\boldsymbol{i}} - \bar{D}\boldsymbol{\alpha_i}\|_2^2 + \lambda\|\boldsymbol{\alpha_i}\|_1. \tag{2.9}$$

This joint sparse coding scheme can only be optimal in the concatenated feature space of $\mathcal{X}$ and $\mathcal{Y}$ but not in each space individually. In the decomposition stage only the observation signal $\boldsymbol{y_i}$ is given and we want to recover the corresponding HR patch $\boldsymbol{x_i}$. Therefore there is no possibility to ensure that the found sparse representation vector $\boldsymbol{\alpha_i}$ is optimal in the HR space, $\mathcal{X}$. Due to this shortcoming we developed a bilevel formulation which we explain in detail in the next chapter.

Another state-of-the-art SR algorithm based on sparse coding is the work of Zeyde et al.[ZEP12]. They follow the work of Yang et al.[YWHM10] but make some important modifications. At the preprocessing stage, a dimensionality reduction is performed on the LR features from the LR image making the dictionary training faster. More importantly, they avoid the suboptimal joint SC scheme used in [YWHM10] by training primarily the LR dictionary with the aid of the K-SVD[1] dictionary training developed in [AEB06]. A side product of training the LR dictionary is the sparse representation vector inferred from the LR dictionary. With this SC vector at hand for each training sample, they learn the HR dictionary following equation (2.10). Note that in this training process, the sparse vector is optimal in the LR feature space and the HR dictionary guarantees that the same vector is optimal in the HR signal space. Thus, this training process overcomes the suboptimal training scheme developed in [YWHM10].

$$D_h = \arg\min_{D_h} \sum_{N} \|\boldsymbol{x_i} - D_h\boldsymbol{\alpha_i}\|_2^2. \tag{2.10}$$

In addition they develop a more complex global training scheme for the HR dictionary using a global image based patch extraction operator. This operator, simply a special matrix, extracts all patches of an image and takes the overlap of the high-resolution patches into account. Using such an operator enforces spatial coherence within the training. In the reconstruction an image is split into patches and features are taken. The dimensionality reduction is performed and the Orthogonal Matching Pursuit (OMP) algorithm[RZE08] is applied on the reduced set of LR features utilizing the LR dictionary. The resulting sparse vectors are used to reconstruct HR patches with the aid of the high-resolution dictionary.

[1] Singular Value Decomposition (SVD) algorithm generalizing K-means clustering

The actual HR image is formed by solving a Least-Squares (LS) problem on the difference between the approximated patches and the actual image incorporating the extraction operator. This LS problem has a closed form solution and can therefore be solved efficiently.

An interesting combination of a Sparse Coding and a Neighborhood Embedding approach is the work of Timofte et al.[TDG13] and its further developed version [TDSVG14]. Their system learns two dictionaries, a HR and a LR dictionary, and regressors anchored to the dictionary atoms. They borrow the dictionary learning method from Zeyde et al.[ZEP12] but use a totally different decomposition approach, rather similar to NE. While normally sparse decomposition follows equ. (2.5) where the l_1-norm is used as regularization, they instead employ the l_2-norm on the sparse coefficient vector resulting in a Ridge Regression (RR)[TA77] which has a closed-form solution. In the global case, meaning all dictionary atoms are used as neighbors to the input feature, this leads to a projection matrix that can be precomputed and is given by

$$\begin{aligned} \boldsymbol{x} &= D_h(D_l^T D_l + \lambda I)^{-1} D_l^T \boldsymbol{y_F}, \\ P_G &= D_h(D_l^T D_l + \lambda I)^{-1} D_l^T, \end{aligned} \tag{2.11}$$

where $\boldsymbol{x}$ is the HR patch, $\boldsymbol{y_F}$ the LR input feature, D_l and D_h the low- and high-resolution dictionary, respectively and P_G is the global projection matrix. As this formulation is very general, they propose to group the dictionary atoms into neighborhoods based on the correlation between atoms rather then the Euclidean distance. Once the neighborhood of the atoms is defined, they detachedly pre-compute the projection matrix P_j for each atom $\boldsymbol{d_j}$ of the dictionaries utilizing their neighbors. This can all be calculated offline and in advance. The actual SR problem can then be solved by finding the nearest dictionary atom $\boldsymbol{d_j}$ to the input feature $\boldsymbol{y_{iF}}$ in the LR dictionary and use the associated projection matrix P_j to map the input feature to the HR space. One can imagine that this method can be computed efficiently and has a fast runtime since only an NN search has to be solved and no optimization is needed.

In figure 2.6 we show SR estimates of different sparse coding SR methods. We compare the results of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13]. By inspecting the image details of each method, one can see slight differences in the quality. As the image of Yang et al. shows, this method can not super resolve textures and image details as well as the others and their results are very smooth. The superior methods of Zeyde et al. and Timofte et al. produce more realistic HR images but do not much differ from each other in terms of Image quality assessment (IQA).

(a) original (b) Yang et al. 25.1dB

(c) Zeyde et al. 25.4dB (d) Timofte et al. 25.4dB

(e) original cu (f) Yang cut (g) Zeyde cut (h) Timofte cut

Figure 2.6.: Result of SC SR algorithms upsampled by a magnification factor of 3. First image shows the original input file “barbara”. The following images from left to right and top to bottom show SR results of [YWHM10](25.1dB), [ZEP12](25.4dB) and [TDG13](25.4dB), respectively. One can see that [YWHM10] gives a slightly smoother result, while [ZEP12] and [TDG13] can resolve more realistic images.

2.6. Approximation of the l_1-norm

Before we introduce the bilevel optimization procedure, it is important to clarify some basic properties of the l_1-norm and the approximation we are using, the $l_{1,\epsilon}$-norm. The l_1-norm is a common regularization in convex optimization. In the context of SC it is used as an alternative to the l_0 pseudo-norm, which is a non-convex semi-norm counting the non-zero components in a vector. A regularization using the l_0-norm is utilized for giving sparse solution vectors. Likewise, the l_1-norm is a sparsity inducing norm[BJMO12] and such a property is inherent to SC. In our first algorithm we incorporate a smooth approximation of the l_1-norm, the $l_{1,\epsilon}$-norm, with its derivations given by

$$\Phi(\boldsymbol{x}) = \sqrt{\boldsymbol{x}^2 + \epsilon^2}, \tag{2.12}$$

$$\Phi'(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\sqrt{\boldsymbol{x}^2 + \epsilon^2}}, \tag{2.13}$$

where $\boldsymbol{x}$ is the sparse vector and ϵ is a small scalar constant. The major benefit of using this approximation is, that it is infinitely often differentiable. From a numerical point of view, regularization with the $l_{1,\epsilon}$-norm should lead to equal results while having the advantage of being differentiable and it can be applied while disregarding additional assumptions. In contrast to the $l_{1,\epsilon}$-norm, the first order derivative of the l_1-norm can only be evaluated at point $x \neq 0$ and is given by

$$\frac{\mathrm{d}|x|}{\mathrm{d}x} = \frac{x}{|x|}, \quad \forall x \neq 0. \tag{2.14}$$

The sub-differential formula is given by

$$\frac{\partial |x|}{\partial x} = \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x < 0 \\ [-1, 1] & \text{else.} \end{cases} \tag{2.15}$$

The second derivative of $|x|$ with respect to x is zero everywhere except at point zero, where it does not exist.

Figure 2.7(a) shows the absolute value function, $|x|$ compared to its approximation equ. (2.12) and figure 2.7(b) shows the derivative of the $l_{1,\epsilon}$-norm compared to the subdifferential of the l_1-norm. Note that in our implementation ϵ is set to

10^{-6} but for presentation we set it to a higher value.

2.7. The Fast Iterative Shrinkage/Thresholding Algorithm

At this point we want to show the basic properties of the $l_{1,\epsilon}$-regularization when incorporated in the FISTA. The FISTA belongs to the first-order convex optimization methods which only use the objective value and the (sub)gradient to optimize functions. The FISTA is an accelerated version of the rather slow-converging group of Iterative Shrinkage-Thresholding Algorithms (ISTAs). It can be used to tackle unconstrained minimization problems of a sum of two convex function $f(x)$ and $g(x)$, given by

$$\min_{x} f(x) + \lambda g(x). \tag{2.16}$$

The LASSO problem or the sparse decomposition problem stated in (2.5) are examples of such problems. We recall (2.5) given by

$$\min_{\boldsymbol{x}} \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{x}\|_1. \tag{2.17}$$

Any functions satisfying following requirements can be solved by FISTA/ISTA.

1. $f + g$ admits a minimizer x^*
2. f is convex, smooth and differentiable
3. g is convex, subdifferentiable and simple[2]

To understand FISTA we recall the standard procedure of ISTA. ISTA splits the optimization problem, making a gradient step of the smooth function f and applying the proximal map on the result of the gradient step solving the non-smooth function g. For problem (2.17) the general gradient step of ISTA is given by

$$\boldsymbol{x}_{k+1} = \mathcal{T}_{\lambda\tau}\left(\boldsymbol{x}_k - \eta A^T(A\boldsymbol{x}_k - \boldsymbol{b})\right) \tag{2.18}$$

[2]the prox-map has a closed-form solution or can be rapidly solved numerically

where η is the step size and $\mathcal{T}_{\lambda\tau} : \mathbb{R}^n \to \mathbb{R}^n$ is the shrinkage or proximal operator of the l_1-norm defined by

$$\mathcal{T}_{\lambda\tau}(\boldsymbol{x}_i) = (|\boldsymbol{x}_i| - \lambda\tau)_+ \operatorname{sgn}(\boldsymbol{x}_i). \tag{2.19}$$

The step size of both algorithms depends on the *Lipschitz* constant of ∇f, for (2.17) it is given by

$$\begin{aligned} L(f) &= \lambda_{max}(A^T A) \\ \eta &\leq \frac{2}{L} = \frac{2}{\|A^T A\|}. \end{aligned} \tag{2.20}$$

In comparison to ISTA, FISTA applies the shrinkage operator not on the gradient step of f directly but rather at a very specific linear combination of the previous two iterates of $\boldsymbol{x}$ resulting in the increased rate of convergence. In algorithm 1 we give a summary of the FISTA.

Algorithm 1 Summary of the FISTA

Require: input $A, \boldsymbol{b}, \lambda$, and for a hot-start $\boldsymbol{x}_0$
  $\eta leq \frac{1}{\|A^T A\|}$, $\boldsymbol{y}_1 = \boldsymbol{x}_0 \in \mathbb{R}^n$ and $t_1 = 1$
  while not converged **do**
    $\boldsymbol{x}_{k+1} = \mathcal{T}_{\lambda\tau}\left(\boldsymbol{y}_k - \eta A^T(A\boldsymbol{y}_k - \boldsymbol{b})\right)$
    `/* with` $\mathcal{T}_{\lambda\tau}(x)$ `given by` (2.19) `*/`
    $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
    $\boldsymbol{y}_{k+1} = \boldsymbol{x}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$
  end while

Algorithms using the proximal operator to solve convex optimization problems are called proximal algorithms. They are well suited for non-smooth, constrained and large scale problems especially if the proximal operator can be evaluated sufficiently[PB14]. The proximal algorithm adds a quadratic function to the objective, transforming it to a strongly convex function even if the objective is non-smooth. Let $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a closed proper convex function, then the *proximal-operator* $\mathbf{prox}_g : \mathbb{R}^n \to \mathbb{R}^n$ of g is defined by

$$\mathbf{prox}_g(y) := \arg\min_x \left(g(x) + \frac{1}{2}\|x - y\|_2^2 \right). \tag{2.21}$$

Equation (2.21) states the unscaled prox-operator and has a unique minimizer for every $y \in \mathbb{R}^n$. In most cases, as in our own, we have a scaled prox-operator. The scaling is done by adding a scalar parameter to the 1/2-term, giving $1/(2\tau)$. In the scaled version, τ plays a role similar to the step-size parameter in gradient methods. By inspecting the *proximal-operator* of the l_1-norm we see the well-known soft-threshold operator as stated in (2.19). The smooth approximation, the $l_{1,\epsilon}$-norm (2.12), results in a different *proximal-operator*. In this case the prox-operator point-wise solves a quadric polynomial equation given by,

$$\mathbf{prox}_{\lambda\tau g}(y) := \arg\min_{x} \frac{1}{2\tau}(x-y)^2 + \lambda\sqrt{x^2+\epsilon^2}. \tag{2.22}$$

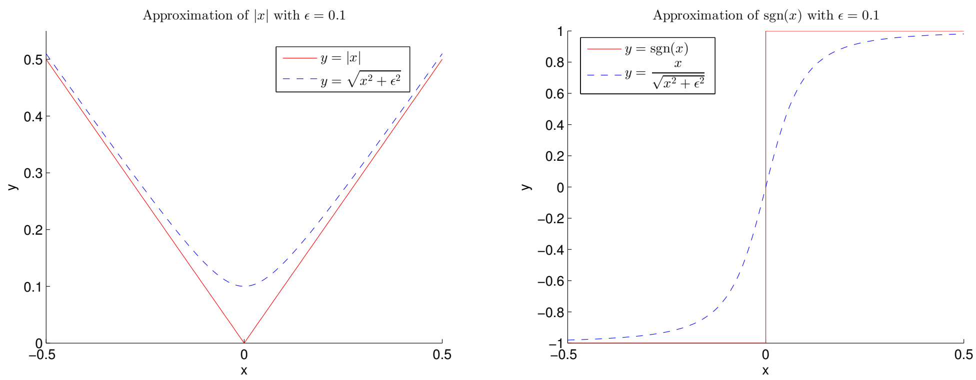
This prox-operator cannot be solved explicitly in reasonable time, thus we apply Newton's method[Wei14] to solve the quadric equation. This leads us to the following derivations of (2.22) stated point-wise as,

$$\begin{aligned} f': \quad 0 &= (x-\hat{x})\sqrt{x^2+\epsilon^2} + \tau\lambda x, \\ f'': \quad 0 &= \frac{x(x-\hat{x})}{\sqrt{x^2+\epsilon^2}} + \sqrt{x^2+\epsilon^2} + \tau\lambda, \\ x^{n+1} &= x^n - \frac{f'}{f''}. \end{aligned} \tag{2.23}$$

Newton's method converges in very few iterations and can be evaluated efficiently. Figure 2.8 show the soft-thresholding operator obtained by solving the proximal algorithm on the scaled l_1-norm and compares it to the prox-operator of the smoothed scaled $l_{1,\epsilon}$-norm.

2.8. The Inertial Proximal Algorithm For Strongly Convex Optimization

A newly presented algorithm called Inertial Proximal Algorithm for strongly convex Optimization (IPIASCO)[OBP14] can solve strongly convex optimization problems of certain type with an even better convergence rate than FISTA or equivalent algorithms. It makes the same assumptions as FISTA given in (2.16) yet surpasses the optimal rate of convergence for f or g being strongly convex and f being twice differentiable. Fortunately, the problem of (2.17) combined with the $l_{1,\epsilon}$-regularization poses such a problem whereby a linear convergence rate can be



(a) $|x|$ compared to $\Phi(x)$ (b) $\mathrm{sgn}(x)$ compared to $\Phi'(x)$

Figure 2.7.: Approximation of the l_1-norm of a variable x and its derivative. Figure (a) shows the absolute value function $|x|$ and our approximation, $\Phi(x)$ defined in equ. (2.12). Figure (b) shows the derivatives, function $\mathrm{sgn}(x)$ compared to the derivative $\Phi'(x)$.

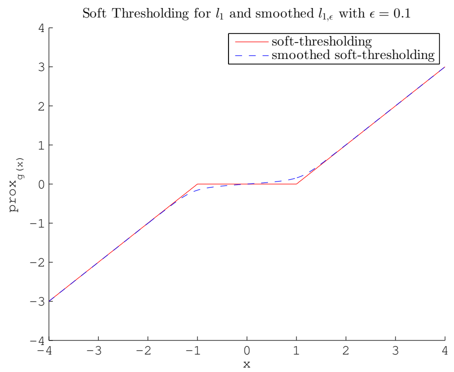


Figure 2.8.: This figure shows the solution of the prox-operators. Red shows the standard shrinkage thresholding function used by FISTA compared to the resulting prox-operator of the $l_{1,\epsilon}$-norm.

achieved. The IPIASCO exploits the structure of strongly convex functions utilizing the Lipschitz-constants and the convexity parameters. With these parameters the algorithm is able to adapt the step size such that an increased convergence rate can be achieved. The general gradient step of IPIASCO follows the heavy-ball method[Pol87] and is given by

$$x_{n+1} = (I + \alpha\partial g)^{-1} \left(x_n - \alpha\nabla f + \beta(x_n - x_{n-1})\right). \tag{2.24}$$

where α and β are specifically chosen step size parameters and $(I + \alpha\partial g)$ is the prox-operator of g. The step size parameters are given by

$$\alpha = \frac{4}{(\sqrt{l+m} + \sqrt{L+m})^2 - 4m}, \beta = \frac{(\sqrt{m+L} - \sqrt{m+l})^2}{(\sqrt{m+L} + \sqrt{m+l})^2 - 4m} \tag{2.25}$$

where L is the Lipschitz constant of ∇f and l and m are the convexity parameters of f and g , respectively. In our case $l = 0$ and $m = \min_{x \in [lb,ub]} \lambda \frac{\epsilon^2}{(\boldsymbol{x}^2+\epsilon^2)^{\frac{2}{3}}}$. The Lipschitz constant of ∇f is the same as that in FISTA. The IPIASCO is summarized in algorithm 2.

Algorithm 2 Summary of the IPIASCO

Require: input $A, \boldsymbol{b}, \lambda$, and for a hot-start $\boldsymbol{x}_0$

$L = \frac{1}{\|A^T A\|}$, /* α and β is given by (2.25)*/

while not converged **do**

$\boldsymbol{x}_{k+1} = \prod_C \left(\boldsymbol{x}_k - \alpha A^T(A\boldsymbol{x}_k - \boldsymbol{b}) + \beta(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right)$

/* with $\prod_C(\boldsymbol{x})$ given by (2.22) */

end while

2.9. Bilevel Optimization for Coupled Feature Spaces

The main contribution of this work is a bilevel optimization algorithm extending the dictionary learning problem to coupled feature spaces. The bilevel program enables us to learn the dictionaries hierarchically and ensures the goal that both dictionaries are optimal in each space while having a common sparse representation. Analogous procedures have been developed by Yang et al. in [YWL+12a] and [YWL+12b]. As previously stated, the dictionary learning problem for coupled feature spaces should be formulated such that the dictionaries are optimal in both feature spaces individually. We recall that a bilevel program is a hierarchical optimization problem as they contain a nested optimization problem within the constraint of another optimization problem[Dem02]. Given a common sparse representation, we can easily argue that the dictionary learning problem can be modeled hierarchically such that an optimal LR dictionary in the LR feature space is a requirement to optimize the HR dictionary in the HR feature space since in decomposition only the LR feature space is given. In our case we formulate the bilevel program such that it minimize the error in the high-resolution feature space, while requiring an optimal solution of the sparse decomposition in the low-resolution feature space. Therefore the bilevel program guarantees that the found sparse representation selecting the dictionary atoms is optimal in the LR features space and in the HR feature space. But more important the bilevel program can propagate the error found in the high-resolution feature space to a change of the low-resolution dictionary and the high-resolution dictionary such that this error is minimized.

Zeyde et al.[ZEP12] in contrast only learn an optimal LR dictionary. Subsequently they use the corresponding HR training set and the sparse representation found in the LR feature space to create a high-resolution dictionary. Their approach is not capable to change the low-resolution dictionary or the sparse representation due to errors in the high-resolution feature space.

Another advantage of the bilevel formulation is that the mapping function connecting the two feature spaces does not need to be known as this is inherently formulated in the bilevel program. This is beneficial to our system since we select the first and second order central difference features in the LR feature space but use the HR patches directly in the HR feature space. The mapping function connecting the two features spaces could still be formulated as a linear function but we do not need to model it.

2.10. Linear Algebra and Matrix Differentiation

Before we begin with the actual bilevel optimization problem statement and derivation we want to recall some basic properties of matrix calculus since we need them later on. Differentiation of a matrix function $F(X)$ in regard to a matrix X is not as straight forward as one might think. Several different notations exist, each of which have their own justifications, however we will only recall the notation we use. The interested reader is referred to [MN99] for further details. From vector calculus we know that if $f(\boldsymbol{x})$ is an $m \times 1$ vector function of an $n \times 1$ vector $\boldsymbol{x}$, then the derivative or *Jacobian* matrix of f in respect to $\boldsymbol{x}$ is a $m \times n$ matrix,

$$\mathrm{D}\, f(\boldsymbol{x}) = \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}^T}. \tag{2.26}$$

Generalizing this formulation to matrix functions of matrices admits the following definition.

Definition 2.1. Let F be a differentiable $m \times n$ real matrix function of the real valued matrix X with size $p \times q$, then the *Jacobian* matrix of F at X is the $mn \times pq$ matrix

$$\mathrm{D}\, F(X) = \frac{\partial \operatorname{vec} F(X)}{\partial (\operatorname{vec} X)^T}\,. \tag{2.27}$$

With this notation one guarantees that all properties of the *Jacobian* matrix are preserved. Furthermore the study of matrix functions of matrices is reduced to the study of vector functions of vectors. The gradient of the matrix function $F(X)$ is given by transposing the *Jacobian* matrix $\mathrm{D}\, F(X)$, hence $\nabla F(X) = \mathrm{D}\, F(X)^T$.

After having a definition for deriving matrix functions, we recall some useful linear algebra notations. These will be employed in the next chapter. First we bring the Kronecker product in mind and the relation to the vec-operator. Let A be a matrix of size $m \times n$ and B be a matrix of size $p \times q$. The $mp \times nq$ matrix defined by

$$\begin{pmatrix} a_{1,1}B & \cdots & a_{1,n}B \\ \vdots & \ddots & \vdots \\ a_{m,1}B & \cdots & a_{m,n}B \end{pmatrix} \tag{2.28}$$

is called the Kronecker product of A and B and is written as $A \otimes B$. Note that the matrix product of AB is only defined if the numbers of columns of A is equal the number of rows in B, hence $n = q$. The Kronecker product in comparison is defined for any pair of matrices. Transposing a Kronecker product gives

$$(A \otimes B)^T = (A^T \otimes B^T). \tag{2.29}$$

Assume we have a valid matrix product AXC then

$$\operatorname{vec}(AXC) = (C^T \otimes A) \operatorname{vec}(X). \tag{2.30}$$

The proof of this theorem is left out here. We refer to [MN99, p.32] for details. A special case of (2.30) is the vector function $AX\boldsymbol{c}$, where $\boldsymbol{c}$ is a vector, then

$$AX\boldsymbol{c} = (\boldsymbol{c}^T \otimes A)\operatorname{vec}(X)\,. \tag{2.31}$$

Furthermore we recall the commutation matrix K_{mn}. Let A be a $m \times n$ matrix, then $\operatorname{vec}(A)$ and $\operatorname{vec}(A^T)$ have the same mn components but their entries are in a different order. Thus there exists a unique $mn \times mn$ permutation matrix which transforms $\operatorname{vec}(A)$ into $\operatorname{vec}(A^T)$. This matrix is called the commutation matrix K_{mn}. Hence

$$K_{mn}\operatorname{vec}(A) = \operatorname{vec}(A^T). \tag{2.32}$$

The matrix K_{mn} is orthogonal, hence

$$K_{mn}^T = K_{mn}^{-1} = K_{nm}. \tag{2.33}$$

Concluding we define A as a $m \times n$ matrix and $\boldsymbol{b}$ as $p \times 1$ vector. Then

$$K_{mp}(\boldsymbol{b} \otimes A) = (A \otimes \boldsymbol{b}). \tag{2.34}$$

Again, proof to this equations is given in [MN99, p.55] and is omitted at this point.

2.11. Summary

In this chapter we have introduced some major single image SR methods and algorithms. Among these categories there is a lot of ongoing research making it difficult to give a comprehensive summary. Furthermore we have given a short overview of convex optimization algorithms. These algorithms are utilized to solve the bilevel programs defined in the next chapter. We concluded this chapter by giving some tools for matrix differentiation that are also applied in the bilevel optimization procedure.

3. Bilevel Optimization for Dictionary Learning Problems

Contents

In this chapter we formulate a bilevel sparse coding program with smoothed $l_{1,\epsilon}$-regularization for the dictionary learning problem and describe the benefits and drawbacks of the $l_{1,\epsilon}$-regularization. We further develop a derivation of this program facilitating the common l_1-regularization. This leads to an algorithm which uses only the active set of dictionary atoms while omitting all other atoms. We conclude this chapter with a discussion of the benefits and disadvantages of these two dictionary learning algorithms.

3.1. Bilevel Program with Smoothed $l_{1,\epsilon}$-Regularization

Bilevel optimization belongs to the class of hierarchical mathematical programs and is closely related to mathematical programs with equilibrium constraints [CMS07]. The major feature of bilevel programs is that they include two mathematical programs in a single instance and one of these programs is part of the other's constraint. In the general setup a bilevel program consists of an upper-level problem and a lower-level problem both of which can have constraints associated. Therefore a bilevel program tries to find the optimal solution for both, the lower-level and the upper-level problem, even if they have opposite objectives.

In our case we have two closely related dictionary training problems, one on the low-resolution features and one on the high-resolution patches, both sharing the same sparse vector. Our bilevel program optimizes both dictionaries simultaneously. The definition of our bilevel program is given as

$$L\{\boldsymbol{\alpha}, D_l, D_h\} = \min_{\boldsymbol{\alpha}, D_l, D_h} \quad \sum_{i=1}^{N} \|D_h\boldsymbol{\alpha}_i - \boldsymbol{f^h}_i\|_2^2 \tag{3.1a}$$

$$s.t. \qquad E\{\boldsymbol{\alpha}_i\} = \arg\min_{\boldsymbol{\alpha}_i} \frac{1}{2}\|D_l\boldsymbol{\alpha}_i - \boldsymbol{f^l}_i\|_2^2 + \lambda\Phi(\boldsymbol{\alpha}_i) \tag{3.1b}$$

where $\boldsymbol{\alpha}_i$ are the sparse vectors, $\boldsymbol{f^h}_i$ and $\boldsymbol{f^l}_i$ are the high and low-resolution patches pairs, D_h and D_l the high and low-resolution dictionaries respectively, and $\Phi(\boldsymbol{\alpha})$ is the smoothed $l_{1,\epsilon}$-norm stated in equ. (2.12). The variable $\boldsymbol{f^h}_i$ is a $k \times 1$ vector, D_h is an $k \times n$, D_l an $m \times n$ matrix, $\boldsymbol{\alpha}_i$ an $n \times 1$, $\boldsymbol{f^l}_i$ an $m \times 1$ vectors. In comparison to [YWL$^+$12b] we do not have a norm constraint on the dictionary atoms; $D(:,k) \leq 1$. The main reason why the norm constraint is employed, is to prevent the trivial solution of infinitely large dictionary atoms and infinitely small sparse vectors $\boldsymbol{\alpha}$. We believe that our training scheme implicitly learns the correct norm of each atom because we are also initializing the program with dictionaries trained by[YWHM10].

By inspecting the structure of our bilevel optimization problem (3.1) we see that the lower-level objective is a strictly convex function without any constraint added. This implies that we can find a unique minimizer $\boldsymbol{\alpha}^*$. The upper-level objective states a linear program. A bilevel program with such a structure can be solved by reformulating it as a single-level optimization problem. This is done by deriving the optimality condition of the lower level-objective (3.1b) and adding it as a constraint to the upper-level objective (3.1a). The resulting constraint optimization problem can then be rewritten as a unconstrained optimization problem introducing a Lagragian multiplier associated with the constraint. This newly created Lagragian function can then be solved by differentiation in regard to all unknown variables and eliminating the Lagragian multipliers and sparse vector. This leads to the derivatives of the upper-level objective (3.1) with regard to the dictionaries D_h and D_l, while keeping the optimal $\boldsymbol{\alpha}$ inferred from the lower-level objective(3.1b). In our algorithm we subsequently plugin the derivatives in the quasi-Newton method of the Limited Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm[LN89]. The same derivatives could be found by differentiation of the upper-level objective where the chain rule would be applied followed by implicit differentiation of $\boldsymbol{\alpha}$ in respect to D_l. This procedure was employed by Yang et al. in [YWL+12b].

In our formulation of the bilevel program(3.1) we use the smoothed approximation to the l_1-norm, the $l_{1,\epsilon}$-norm. The obvious advantage of using the $l_{1,\epsilon}$-regularization in the lower-level objective is that it is continuously differentiable and strictly convex. This means that the first-order optimality condition is sufficient for global optimality and a unique minimizer can be found. Furthermore this functional is twice differentiable at all points. In comparison, if the l_1-regularization in the lower-level objective is employed we would also reach a global but not necessarily unique minimizer. Additionally the l_1-norm has no second-order derivative at point zero requiring supplemental assumptions if the same algorithm is applied. The first-order necessary optimality condition of (3.1b), which is also sufficient, is calculated by deriving it with respect to $\boldsymbol{\alpha}$ and setting it to zero, giving,

$$\left.\frac{\partial E}{\partial \boldsymbol{\alpha}}\right|_{\boldsymbol{\alpha}^*(D_l)} = D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i + \lambda \frac{\boldsymbol{\alpha}_i}{\sqrt{\boldsymbol{\alpha}_i^2 + \epsilon^2}} = 0. \tag{3.2}$$

This equation is also referred to as the stationary condition of the lower-level objective. We now add the stationary condition (3.2) as a constraint to the upper-level problem (3.1a). The resulting single level constraint optimization problem is given by

$$\begin{aligned} L\{\boldsymbol{\alpha}, D_l, D_h\} = \min_{\boldsymbol{\alpha}, D_l, D_h} \quad & \sum_{i=1}^{N} \|D_h \boldsymbol{\alpha}_i - \boldsymbol{f^h}_i\|_2^2 \\ s.t. \quad & \nabla_{\boldsymbol{\alpha}} E\{\boldsymbol{\alpha}_i\} = D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i + \lambda \frac{\boldsymbol{\alpha}_i}{\sqrt{\boldsymbol{\alpha}_i^2 + \epsilon^2}} = 0. \end{aligned} \tag{3.3}$$

Since this equation(3.3) states an optimization problem with equality constraint, it can easily be reformulated as an unconstrained optimization problem with the aid of *Lagrangian* multipliers. The reformulated unconstrained upper-level objective is given by

$$\max_{\boldsymbol{p}_i} \min_{\boldsymbol{\alpha}, D_l, D_h} \quad \sum_{i=1}^{N} \|D_h \boldsymbol{\alpha}_i - \boldsymbol{f^h}_i\|_2^2 + \left\langle \boldsymbol{p}_i, D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i + \lambda \frac{\boldsymbol{\alpha}_i}{\sqrt{\boldsymbol{\alpha}_i^2 + \epsilon^2}} \right\rangle , \tag{3.4}$$

where $\boldsymbol{p}_i$ are the *Lagrangian* multipliers. This equation can now be derived with respect to the variables $\boldsymbol{p}, \alpha, D_l$ and D_h. For the derivation in respect to D_h we swap D_h and $\boldsymbol{\alpha}$ in the matrix-vector product $\|D_h \boldsymbol{\alpha}_i - \boldsymbol{f^h}_i\|$ to $\|K_i \operatorname{vec}(D_h) - \boldsymbol{f^h}_i\|$ where the matrix K_i is the reordered $\boldsymbol{\alpha}_i$ vector of size $k \times n * k$ given by $K_i = (\boldsymbol{\alpha}_i^T \otimes I_k)$. The remaining derivatives are given as,

$$\frac{\partial L}{\partial \boldsymbol{p}_i} = D_l^T D_l + \lambda \operatorname{diag} \frac{\epsilon^2}{(\boldsymbol{\alpha}_i^2 + \epsilon^2)^{\frac{3}{2}}} = 0, \tag{3.5}$$

$$\frac{\partial L}{\partial \boldsymbol{\alpha}_i} = D_h^T D_h \boldsymbol{\alpha}_i - D_h^T \boldsymbol{f^h}_i + \boldsymbol{p}_i \left(D_l^T D_l + \lambda \operatorname{diag} \frac{\epsilon^2}{(\boldsymbol{\alpha}_i^2 + \epsilon^2)^{\frac{3}{2}}} \right), \tag{3.6}$$

$$\frac{\partial L}{\partial D_l} = \boldsymbol{p}_i \frac{\partial \left(D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i \right)}{\partial D_l}, \tag{3.7}$$

$$\frac{\partial L}{\partial D_h} = {K^T}_i (K_i \operatorname{vec}(D_h) - \boldsymbol{f^h}_i). \tag{3.8}$$

Equation (3.6) can now be solved explicitly in respect to the Lagrangian $\boldsymbol{p}$ and inserted in (3.7) following,

$$\begin{aligned} \boldsymbol{p} &= - \left(\frac{\partial L}{\partial \boldsymbol{\alpha}} \right) \left(\frac{\partial^2 E}{\partial \boldsymbol{\alpha}^2} \right)^{-1} \\ \Rightarrow \boldsymbol{p}_i &= - \left(D_h^T D_h \boldsymbol{\alpha}_i - D_h^T \boldsymbol{f^h}_i \right) \left(D_l^T D_l + \lambda \ \operatorname{diag} \frac{\epsilon^2}{(\boldsymbol{\alpha}_i^2 + \epsilon^2)^{\frac{3}{2}}} \right)^{-1}, \\ \frac{\partial L}{\partial D_l} &= - \left(D_h^T D_h \boldsymbol{\alpha}_i - D_h^T \boldsymbol{f^h}_i \right) \left(D_l^T D_l + \lambda \operatorname{diag} \frac{\epsilon^2}{(\boldsymbol{\alpha}_i^2 + \epsilon^2)^{\frac{3}{2}}} \right)^{-1} \\ & \quad \frac{\partial \left(D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i \right)}{\partial D_l}. \end{aligned} \tag{3.9}$$

The partial derivative $\partial \left(D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i \right)$ in respect to ∂D_l is calculated following [MN99] with definition 2.1 and the use of the *vec*-operator and the *Kronecker* product giving,

$$
\begin{aligned}
\mathrm{D}_{D_l} f(D_l) &= \frac{\partial \left(D_l^T D_l \boldsymbol{\alpha} - D_l^T \boldsymbol{f^l} \right)}{\partial vec(D_l)^T} \\
\partial f(D_l) &= \partial(D_l^T D^T \boldsymbol{\alpha}) - \partial(D_l^T \boldsymbol{f^l}) \\
\partial f(D_l) &= I_n \partial(D_l^T) D_l \boldsymbol{\alpha} + D_l^T \partial(D_l) \boldsymbol{\alpha} - I_n \partial(D_l^T) \boldsymbol{f^l} \\
\partial f(D_l) &= (\boldsymbol{\alpha}^T D_l^T \otimes I_n) \partial \operatorname{vec}(D_l^T) + (\boldsymbol{\alpha}^T \otimes D_l^T) \partial \operatorname{vec}(D_l) - (\boldsymbol{f^l}^T \otimes I_n) \partial \operatorname{vec}(D_l^T) \\
\partial f(D_l) &= (\boldsymbol{\alpha}^T D_l^T \otimes I_n) K_{mn} \partial \operatorname{vec}(D_l) + (\boldsymbol{\alpha}^T \otimes D_l^T) \partial \operatorname{vec}(D_l) - (\boldsymbol{f^l}^T \otimes I_n) K_{mn} \partial \operatorname{vec}(D_l) \\
\mathrm{D}_{D_l} f(D_l) &= (\boldsymbol{\alpha}^T D_l^T - \boldsymbol{f^l}^T \otimes I_n) K_{mn} + (\boldsymbol{\alpha}^T \otimes D_l^T) \\
\nabla_{D_l} f(D_l) &= \left((\boldsymbol{\alpha}^T D_l^T - \boldsymbol{f^l}^T \otimes I_n) K_{mn} \right)^T + (\boldsymbol{\alpha}^T \otimes D_l^T)^T \\
\nabla_{D_l} f(D_l) &= K_{nm} (D_l \boldsymbol{\alpha} - \boldsymbol{f^l} \otimes I_n) + (\boldsymbol{\alpha} \otimes D_l) \\
\nabla_{D_l} f(D_l) &= \left(I_n \otimes (D_l \boldsymbol{\alpha}_i - \boldsymbol{f^l}_i) \right) + (\boldsymbol{\alpha}_i \otimes D_l) \,. \qquad (3.10)
\end{aligned}
$$

Note that we have omitted the subscript due to ease of reading except for the result. The derivative of L with respect to D_h and D_l can than be plugged in an LBFGS algorithm[1][LN89][BLNZ95]. This algorithm belongs to the quasi-Newton methods and approximates the second derivatives, the Hessian, of the unknown variables by their previous iterates, in our case D_h and D_l. The algorithm uses rank-one updates specified by gradient evaluation on the unknowns to approximate the Hessian. Beneficially, the LBFGS includes a line search since our bilevel program only gives a decent direction.

As we have now derived the dictionary learning update, we still need the results of the sparse decomposition of the lower-level objective in order to calculate equation (3.9) and (3.8). Thus, we need the optimal α, the unique minimizer α^* of (3.1b). This is a precondition in our bilevel program as we have set the first-order optimality condition of the lower-level objective to zero, as defined in equation (3.2). We have to guarantee that the gradient of the lower-level objective is as small as possible. Since we defined a smoothed $l_{1,\epsilon}$ problem, we cannot use a standard solver. Thus, we decided to use the Inertial Proximal Algorithm for strongly convex Optimization (IPIASCO)[OBP14] to solve the sparse decomposition on the low-resolution dictionary. As the name suggests the IPIASCO is a special solver for strongly convex optimization problems and since the sparse

[1] `http://www.cs.toronto.edu/~liam/lbfgs-1.1.tar.gz`

decomposition with $l_{1,\epsilon}$-regularization poses such a problem, this algorithm fits perfectly for our purpose and converges in linear time. The smoothed $l_{1,\epsilon}$ regularization generates a particular proximity operator sometimes referred as shrinkage operator similar to the famous soft threshold operator in Fast Iterative Shrinkage-Thresholding Algorithm (FISTA). We refer to section 2.6 where we have already defined our proximity operator and IPIASCO.

In our algorithmic settings it is crucial to solve the lower-level objective precisely since our first-order optimality condition demands that the gradient equals zero, $\nabla_\alpha E\{\alpha_i\} = 0$. Only then does the gradient formulation of the upper-level objective become valid and the bilevel program decrease the loss function. Beneficial to the IPIASCO is that we can set the value of the gradient as a convergence criteria. In our implementation we demand that the gradient has to be less than 10^{-7} in order to reach convergence. Algorithm 3 gives a brief summary of our program.

Algorithm 3 Bilevel program with $l_{1,\epsilon}$-regularization solving (3.1)

Require: input $\boldsymbol{f^l}, \boldsymbol{f^h}, \lambda, \epsilon$, initial D_l, D_h
  $x = [\mathrm{vec}(Dl); \mathrm{vec}(Dh)]$
  start LBFGS(x)
  while LBFGS not converged **do**
    /* sparse decomposition with IPIASCO on D_l given $\boldsymbol{f^l}$ */
    /* and $prox_{\Phi(\boldsymbol{\alpha})}$: Newton alg. following (2.23) */
    $\boldsymbol{\alpha} = \mathrm{IPIASCO}(D_l, \boldsymbol{f^l}, \lambda, \epsilon)$
    for all samples $i \in \boldsymbol{f^l}_i$ **do**
      /* calculate derivatives following (3.8) and (3.9) */

$$\begin{aligned}\nabla L_{i_{D_l}} =& - \left(D_h^T D_h \boldsymbol{\alpha}_i - D_h^T \boldsymbol{f^h}_i\right) \left(D_l^T D_l + \lambda \operatorname{diag} \frac{\epsilon^2}{(\boldsymbol{\alpha}_i^2 + \epsilon^2)^{\frac{3}{2}}}\right)^{-1} \\ & \left((I_n \otimes (D_l \boldsymbol{\alpha}_i - \boldsymbol{f^l}_i)) + (\boldsymbol{\alpha}_i \otimes D_l)\right) \\ \nabla L_{i_{D_h}} =& (\boldsymbol{\alpha}_i^T \otimes I_k)\end{aligned}$$

    end for
    $L_{D_l} = \sum_i \nabla L_{i_{D_l}}$
    $L_{D_h} = \sum_i \nabla L_{i_{D_h}}$
  end while

3.1.1. Discussion

Unfortunately, the smoothed $l_{1,\epsilon}$-norm is not truly sparsity inducing. This can be easily seen by inspecting the prox-operator of the $l_{1,\epsilon}$-norm in figure 2.8. Due to this fact the resulting sparse vector $\boldsymbol{\alpha}$ is not sparse anymore. The vector $\boldsymbol{\alpha}$ is instead a full vector with most entries smaller than ϵ. This impacts the runtime of our algorithm, but not to the quality of the result. On the contrary, the results using the smoothed $l_{1,\epsilon}$-norm in the optimization are superior to the results using the l_1-regularization, but the runtime of the training is rather slow. Therefore we developed a simplified training scheme similar to Yang et al. in [YWL+12b] with the l_1-regularization which we present in the next sections.

3.2. Bilevel Program with an Active Set

In sparse decomposition usually only a small set of dictionary atoms are active to describe the input data, meaning most of the dictionary atoms are left out and the coefficient vector $\boldsymbol{\alpha}$ become sparse, for reference see equation (2.5) or (3.1b). This fact can be utilized in the bilevel optimization procedure. The main idea is to apply the same differentiation as we developed earlier but just on the "active atoms" of the dictionary for each training sample. This reduces the computational overhead significantly and faster training can be employed. To apply this simplification we still need to make some assumptions on the l_1-regularization, since the second derivative of the l_1-norm is not defined, at least at point zero. We shortly recapitulate the bilevel program of (3.1) but with the l_1-regularization giving

$$L\{\boldsymbol{\alpha}, D_l, D_h\} = \min_{\boldsymbol{\alpha}, D_l, D_h} \quad \sum_{i=1}^{N} \|D_h \boldsymbol{\alpha}_i - \boldsymbol{f^h}_i\|_2^2 \tag{3.11a}$$

$$s.t. \qquad E\{\boldsymbol{\alpha}_i\} = \arg\min_{\boldsymbol{\alpha}_i} \frac{1}{2}\|D_l \boldsymbol{\alpha}_i - \boldsymbol{f^l}_i\|_2^2 + \lambda \|\boldsymbol{\alpha_i}\|_1. \tag{3.11b}$$

If we inspect the first derivative of the lower-level program (3.11b) and assume that most entries of the coefficient vector α are zero, the problem can be reduced to the set of active dictionary atoms. We recapitulate the first-order sub-differential of the lower-level objective equipped with the l_1-regularization giving,

$$\frac{\partial E}{\partial \boldsymbol{\alpha}_i} = D_l^T D_l \boldsymbol{\alpha}_i - D_l^T \boldsymbol{f^l}_i + \lambda\, \mathrm{sgn}(\boldsymbol{\alpha}_i) = 0. \tag{3.12}$$

At this point we donate Λ_i as the active set of the optimal $\boldsymbol{\alpha^*}_i$ to (3.12), hence $\Lambda_i = \{k : \boldsymbol{\alpha^*}_i(k) \neq 0\}$. Equation (3.12) does not depend on $\boldsymbol{\alpha}_i$, $\forall\, \boldsymbol{\alpha}_i(k) = 0$ and the second derivative with respect to $\boldsymbol{\alpha}_i$ is zero. Further we imply that $\operatorname{sgn}(\boldsymbol{\alpha}_i)$ is constant for all $\boldsymbol{\alpha}_{i_\Lambda}$ and the second derivative of $\operatorname{sgn}(\boldsymbol{\alpha}_i)$ vanishes. Additionally we assume that the chosen dictionary atoms and therefore the non-zero entries in the sparse vector do not change for small perturbations of the dictionary. Yang et al.[YWL+12b] define similar lemmas and gives proof to them. In fact they reach the same derivations as we do.

We can now apply the previous discussed assumptions to the derivations (3.4) and consequentially to (3.8) and (3.9). The derivatives of L with respect to D_{h_Λ} and D_{l_Λ} are given by,

$$\nabla L_{D_l} = -\left(D_{h_\Lambda}^T D_{h_\Lambda} \boldsymbol{\alpha}_{i_\Lambda} - D_{h_\Lambda}^T \boldsymbol{f^h}_i\right)\left(D_{l_\Lambda}^T D_{l_\Lambda}\right)^{-1}\left(\frac{\partial D_{l_\Lambda}^T D_{l_\Lambda}\boldsymbol{\alpha}_{i_\Lambda} - D_{l_\Lambda}^T \boldsymbol{f^l}_i}{\partial D_{l_\Lambda}}\right), \tag{3.13}$$

$$\nabla L_{D_h} = K^T(K \operatorname{vec}(D_{h_\Lambda}) - \boldsymbol{f^h}_i), \tag{3.14}$$

with $K = (\boldsymbol{\alpha}_{i_\Lambda}^T \otimes I_k)$. Compared to equations (3.8) and (3.9), we see only a small difference in the Hessian $\left(D_{l_\Lambda}^T D_{l_\Lambda}\right)^{-1}$ of the lower-level program, where the regularization term has vanished. This is explained by our assumptions on the active set, where we say that $\operatorname{sgn}(\boldsymbol{\alpha}_i)$ is constant and its derivative is zero.

Implementing this algorithm results in faster computation of the derivations since they are only computed on a subset of the dictionary atoms. To do this and make our assumptions valid, we have to compute the sparse decomposition of $\boldsymbol{\alpha}$ on D_l and f_l in advance of each iteration of the LBFGS. We want to clarify that the sparse decomposition in this method needs to be computed with the FISTA algorithm[BT09] and standard soft-thresholding since we have the l_1-regularization in the lower-level program. A brief summary of the active set program is given in algorithm 4.

3.3. Discussion

In this chapter we have derived two comprehensive algorithms through a bilevel program solving the dictionary learning problem for coupled feature spaces. Both of these algorithms exploit the power of bilevel programming and give superior

Algorithm 4 Bilevel program with active set method (3.11)

Require: input $\boldsymbol{f^l}, \boldsymbol{f^h}, \lambda$, initial D_l, D_h

$x = [\mathrm{vec}(Dl); \mathrm{vec}(Dh)]$

/* start lBFGS(x) */

while LBFGS not converged **do**

for all samples $i \in \boldsymbol{f^l}_i$ **do**

`/* sparse decomposition with FISTA on` D_l `given` $\boldsymbol{f^l}_i$ `*/`

$\boldsymbol{\alpha}_i = \mathrm{FISTA}(D_l, \boldsymbol{f^l}_i, \lambda)$

$\Lambda = \{k : \boldsymbol{\alpha}_i(k) \neq 0\}$

`/* calculate derivatives following` (3.14) `and` (3.13) `*/`

$$\begin{aligned}
\nabla L_{i_{D_l}} =& -\left(D_{h_\Lambda}^T D_{h_\Lambda} \boldsymbol{\alpha}_{i_\Lambda} - D_{h_\Lambda}^T \boldsymbol{f^h}_i\right)\left(D_{l_\Lambda}^T D_{l_\Lambda}\right)^{-1} \\
& \left((I_n \otimes (D_{l_\Lambda} \boldsymbol{\alpha}_{i_\Lambda} - \boldsymbol{f^l}_i)) + (\alpha_{i_\Lambda} \otimes D_{l_\Lambda})\right) \\
\nabla L_{i_{D_h}} =& (\boldsymbol{\alpha}_{i_\Lambda}^T \otimes I_k)
\end{aligned}$$

end for

$L_{D_l} = \sum_i \nabla L_{i_{D_l}}$

$L_{D_h} = \sum_i \nabla L_{i_{D_h}}$

end while

testing results compared to the joint training method for coupled feature spaces described in the previous chapter 2.5. The active set method follows the idea of Yang et al.[YWL+12b] and can be computed faster while the smoothed $l_{1,\epsilon}$ regularized bilevel program results in a numerically more stable algorithm. From a numerical point of view the bilevel program with smoothed $l_{1,\epsilon}$-regularization in the lower-level objective is more coherent but also more computationally complex. This is also proven by our evaluations whereby the smoothed $l_{1,\epsilon}$ regularization outperforms the active set method in most cases. We see the reason for this in the better conditioning of the pseudo-inverse of the low-resolution dictionary in equ. (3.13) compared to equ. (3.9). Also the run-time differences are negligible since the training can be performed offline or in advance. Additionally, due to the recently developed IPIASCO[OBP14] and their linear convergence, the run-time of the sparse decomposition performed with IPIASCO is slightly faster compared to FISTA and their results outperform the active set method.

4. Evaluation and Implementation

Contents

In this chapter we give a brief summary of Image quality assessment (IQA) and recapitulate the two previously described algorithms. We show qualitative and objective evaluation and compare our algorithms with state-of-the-art sparse coding super resolution systems. Additionally we give some further details about our implementation.

4.1. Image Quality Assessment

IQA is an active field of research and a number of new methods have been proposed to evaluate image reconstruction systems. Classical qualitative measurements like the Peak-Signal-to-Noise Ratio (PSNR) or the Root Mean Square Error (RMS) error are said to be inconsistent with the human perception because we are much more sensitive to structural errors rather than pure differences in the pixel value. The human eye weights errors on edges or corners higher than in other areas of an image. The major idea behind objective quality measurements as the Structural Similarity (SSIM) index[WBSS04] for example is, to better reflect what people and therefore the human vision defines as a "good". The SSIM index measures the perceived change in the structural information. It is evaluated at a moving window and takes the average, the variance and the dynamic range into account. We evaluate our algorithms with the SSIM-index and the PSNR since PSNR is still the most common qualitative measurement. Other objective IQA metrics include Feature Structural Similarity (FSIM) [ZZMZ11] and Gradient Similarity (GSM)[LLN12] to name just a view.

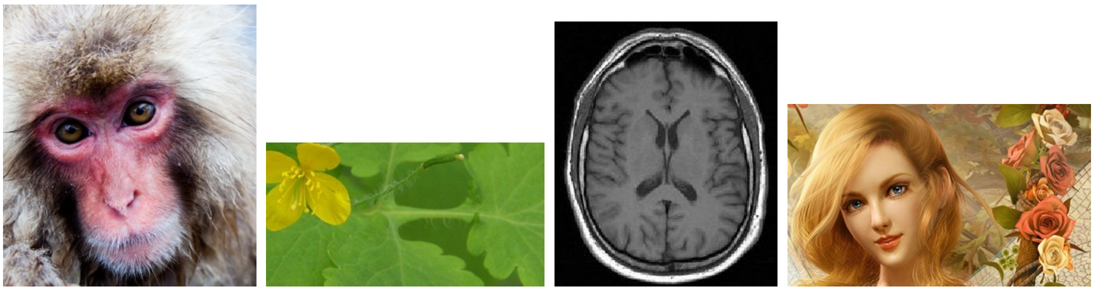
4.2. Dataset

Super Resolution (SR) systems often just evaluate their system on a small set of images and a comprehensive evaluation image database for SR does not exist. The pre-requests for a SR testing database are probably more restrictive than in other fields of image reconstruction. JPEG images for example include already distortion artifacts which would be augmented by SR systems. Often the Kodac Image CD photos[Com99] are used as testing examples. We took a dataset with different classes of images like animals, cars, landscape, buildings, people, flowers, medical images and computer generated graphics. We give credit to Li He[HQZ13] for sending us this comprehensive dataset. We took out one image from each class to train our algorithms and evaluated on all the other images. Figure 4.1 shows our training images. The testing database consists of 72 images, 9 images from each class where one has been taken out for the training. Additionally we trained our algorithm with the images used by Yang et al. in [YWHM10]. Although this dataset only consists of images of flowers, nature images, human faces and cars, this dataset gives equal or even better testing results on both, the testing dataset of Li He and the testing images of Yang et al. We compare our algorithms with the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] as these methods are all based on sparse coding.

4.3. Implementation

The basic points of our implementations regarding the two algorithms, 3 and 4, have already been summarized in the previous chapter. Here we want to give some details about the image preprocessing, the training scheme in general and the sparse decomposition.

(a) car (b) landscape (c) people (d) building



(e) animal (f) flower (g) medical (h) computer graphic

Figure 4.1.: This figure shows our training images. Each image was taken from a class of test images, fig. (a) belongs to cars, (b) to landscapes, (c) to humans, (d) to buildings, (e) to animals, (f) to flowers, (g) to medical images and (h) shows a computer generated image.

(a) face (b) flower (c) nature (d) car

Figure 4.2.: This figure shows some training images from Yang et al.[YWHM10].

4.3.1. Color Treatment

Commonly SR systems only operate on the chroma or luminance channel when processing color images, because the chroma channel comprises the most structural information. Therefore RGB color images are usually transformed to a color space, where a luminance channel is available, in our case the YCbCr color space, and consequently only the luminance channel is processed by the SR system. The remaining color (difference) channels do not contain much structural information and are just bicubically up-sampled for the benefit of faster runtime. Treating color images this way is oriented toward human vision. Color spaces like Lab or YCbCr comprise a separate channel for luminance information as the human eye does by their rod cells in the retina. The cone cells in comparison are less sensitive to light and encode the color information.

4.3.2. Upscaling Factors

A higher upsampling rate is usually achieved by applying a smaller upscaling factor iteratively. For example if a upsampling rate of 4 is desired, the SR pipeline with an upscaling factor of 2 is applied twice, iteratively. If upscaling factors apart from natural numbers are desired the SR system has to be specially trained or upsampled by a higher factor and subsequently downsampled accordingly.

4.3.3. Training Scheme

As for most learning based SR systems, we need training data in both feature spaces and therefore a Low Resolution (LR) and a High Resolution (HR) image pair. In our case we have a system that processes image patches and thus we need patch pairs as training data. For the patch extraction in general and the image preprocessing in particular we choose a similar way as Timofte et al. since it is also based on dictionary learning and currently shows the best qualitative results of dictionary based SR systems.

Imagine we have a LR and HR image pair, they propose to upscale the LR image by a factor of 2 using bicubic interpolation to create a "mid-resolution" image. Then they apply high-pass filters on it and perform a dimensionality reduction using a Principal Component Analysis (PCA). The HR image patches are drawn after subtracting the bicubically upscaled LR image from the HR image. In this manner Timofte et al. learn the difference between the bicubic upsampled LR image and the original HR image patch based on the "mid-resolution" features. We can argue that such a system learns instead a deconvolution rather than an upscaling process. In comparison we do not subtract the bicubically upsampled LR patch from the HR patch, we instead subtract the mean of the "mid-resolution" image patch from the HR image patch. In this manner we learn high-resolution patches independently from their mean and are thus translation invariant regarding the mean of a patch. Therefore our preprocessing consists of four steps. First we take the HR input image and create a downsampled LR image. This LR image is bicubically upsampled to a Mid Resolution (MR) image. This MR image is filtered and patches are drawn. From the unfiltered MR image the mean of each patch is taken and subtracted from the HR patch. The mean-invariant HR patches and the corresponding filtered MR patches then form the training set. We think that dimensionality reduction in the LR feature space is not necessary (although it would lead to a small runtime speedup) because we want to keep as much information as possible about the LR features. The MR image reinforces this objective. First it smooths the LR image and thus features can be drawn without smoothing the kernel. More importantly, the MR image can be seen as a non-linear projection in a higher-dimensional space with similar effects as the kernel-trick in the Support Vector Machine (SVM). The Kernels of the high-pass filters are given by

$$\begin{aligned} K_{1H} &= [-1 \quad 0 \quad 1], \\ K_{1V} &= [-1 \quad 0 \quad 1]^T \\ K_{2H} &= [-1 \quad 0 \quad -2 \quad 0 \quad 1]/2, \\ K_{2V} &= [-1 \quad 0 \quad -2 \quad 0 \quad 1]^T/2 \end{aligned} \tag{4.1}$$

where K_{1H} and K_{2H} are the first and second order central differences in horizontal direction and K_{1V} and K_{2V} are the first and second order central differences in vertical direction, respectively. With this training scheme we learn a LR dictionary composed of MR features and a HR dictionary consisting of patches where the mean has been subtracted. Figure 4.3 shows a semantic overview of our preprocessing and patch extraction scheme.

4.3.4. Norm Constraint on Dictionary Atoms

At this point we want to note that we do not constrain the dictionary columns to have L_2 unit norm. This is done to prevent the trivial solution of infinitely large dictionary atoms and infinitely small sparse vectors α. Since we initialize our dictionaries with the results of a jointly trained dictionary with norm constraints[YWHM10], we think that a L_2 unit-norm constraint is not necessary because the dictionary atoms do not change dramatically. Furthermore we would have to reformulate our model and beyond that, the utilized LBFGS algorithm can not handle an additional reprojection of the dictionaries on the L_2 unit norm. The LBFGS is an algorithm solving unconstrained optimization problems and the reprojection would be a constraint. Note that patches of the same size from images with different spatial resolutions exhibit distinct L_2-norms. The L_2 norm increases with increasing spatial resolution. A goal of our training scheme was to implicitly learn the differences in the norm. This was mainly achieved by omitting the norm constraint on the dictionary atoms. In comparison, Yang et al.[YWHM10] used a norm factor to account for the differences and found this factor by regression. Zeyde et al.[ZEP12] and Timofte et al.[TDG13] chose a different approach by learning only the differences between bicubic upsampled patches and the original HR patches and therefore the norm of the patches becomes insignificant.

4.3.5. Testing Scheme

In the sparse decomposition stage, also referred to as sparse inference or sparse approximation, the LR input image is bicubically upscaled to a MR image where the mean is taken and features of each patch are drawn eventually. The concatenated features are used to perform sparse decomposition on the LR dictionary. The resulting sparse vector α and the HR dictionary are used to form the estimate and the mean of the unfiltered MR patch is added. The features we draw from the MR image are the first and second order gradients given in (4.1). Figure 4.4 shows the preprocessing and the formation of the estimate in the test case.

4.3.6. Remarks on the Patch Size

As we have a patch-based system, it is crucial to take an appropriate patch size for a given upscaling factor. The patch size in combination with the bit depth determines the space of possible patches and scales exponentially with the patch size. Note that the size of a dictionary atom, the squared patch size, and the number of atoms in a dictionary are also correlated. Since a main feature of sparse coding is the over-completeness of their dictionaries, we desire that the dictionary has at least 4-6 times the number of atoms than the size of an atom. Therefore the dictionary size and the patch size are dependent. At this point we want to

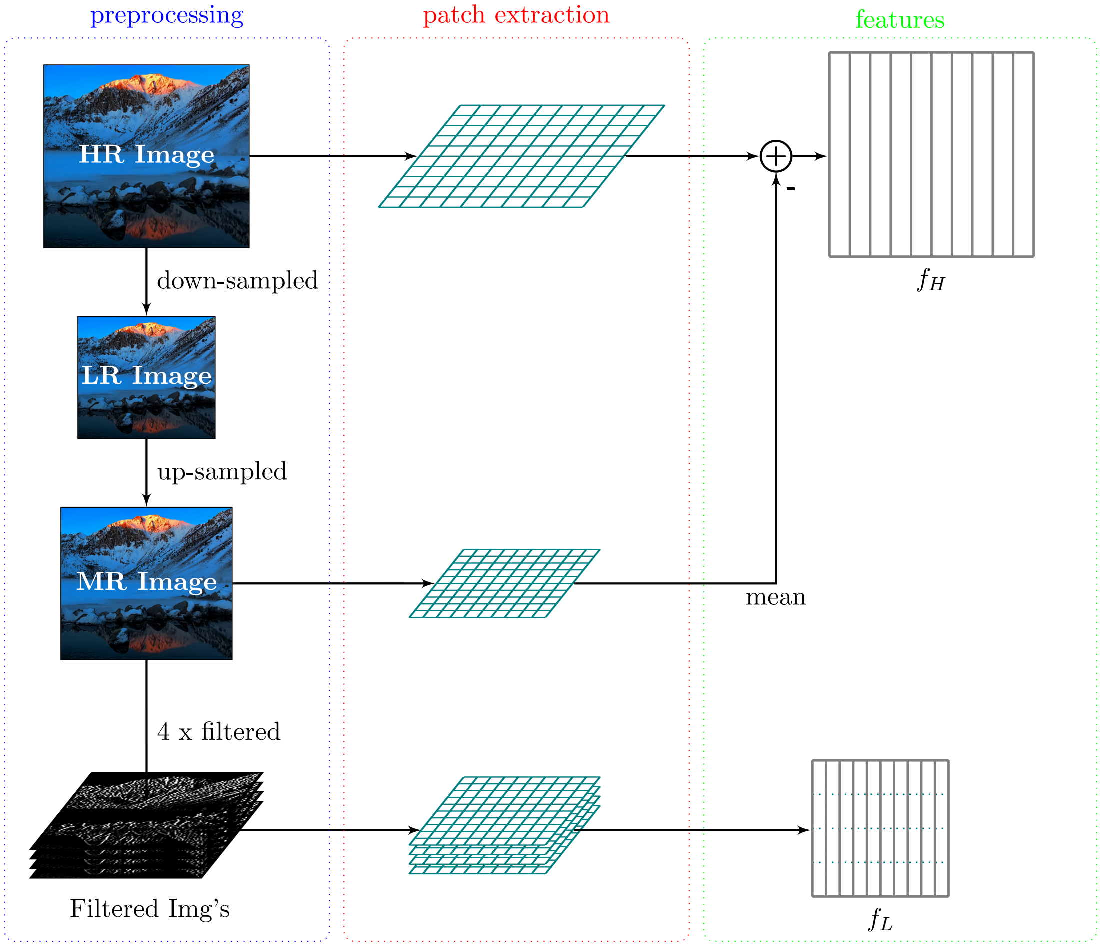


Figure 4.3.: Semantic Overview of the SR training example preprocessing. The image preprocessing can be divided into three section, image preprocessing, patch extraction and feature concatenation. First a HR input image is down-sampled to get a LR input image. The LR image is bicubically upsampled to a "mid-resolution" image. We apply the first and second order central differences filter on this image. Next we extract patches from the high- and mid- resolution images and subtract the mean of a mid-resolution patch from the HR patch. At the same stage we extract patches of the four filtered images and concatenate these feature patches.

remark that the estimates of a patch based system become less meaningful at the boarders. One could desire large dictionaries with high patch size but as the previously stated facts clarify, this has some drawbacks. First of all with a higher patch size the dictionary size grows and this has a large impact on the runtime of the algorithms. Additionally, as the estimates become less meaningful at the boarders we are likely to introduce new error sources. In our experiments we found that a patch size between 6 and 8 pixels for a upscaling factor of 2 grants good results. For a upscaling factor of 3 we used a patch size of 9.

4.3.7. Remarks on the Parameters ϵ and λ

In our first bilevel program developed in the previous chapter, algorithm 3, we used an strongly convex approximation to the l_1-norm, the $l_{1,\epsilon}$-norm. This norm holds a major parameter the ϵ. A basic property of the developed algorithm is the gradient of lower-level objective(3.2), the first order optimality condition, which we seek to be zero. The parameter ϵ is important for reaching this goal. In principle we want ϵ to be as small as possible to better approximate the l_1-norm. The drawback of a small ϵ parameter is the slower convergence of the gradient to reach zero. Apart from this fact, the norm parameter ϵ and the regularization parameter λ are connected. In other words, the ϵ parameter influences the "sparsity" of vector α. For a smaller ϵ, λ has to be smaller too, to get the same number of non-zeros in the sparse vector α. Unfortunately, the $l_{1,\epsilon}$-norm does not truly result in a sparse vector i.e. entries equal to zero. Therefore we can not measure the number of non-zeros. However, we can measure the number of entries higher than a given threshold. Since we do not want to over- or underfit the training, it is important to reach a steady low number of non-zero coefficients. For $e = 10^{-6}$ the number of non-zero coefficients are equal down to a threshold of 10^{-6} which is enough for our purpose and we think that coefficients smaller than 10^{-6} are insignificant. For the test case our major goal is to reach good qualitative estimates in reasonable time and therefore we set the ϵ slightly higher, i.e. $\epsilon = 10^{-5}$.

4.4. Evaluation

We evaluated our algorithms on two datasets, the dataset "Set14" of [TDG13] comprising 14 images and the dataset "Li He" of [HQZ13] containing 72 images. Both test sets where upscaled by magnification factors of 2 and 3. We give objective qualitative measurements in terms of PSNR and SSIM-index and compare the results to the methods of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13]. All measurements have been performed on the luminescent channel, the grayscale of the images, since all compared methods operate on the chroma channel only and the differences are most significant on this channel. For presentation issues we show the resulting RGB images. Additionally we created a real world test set of already degraded image here referred as test set "Mauth-

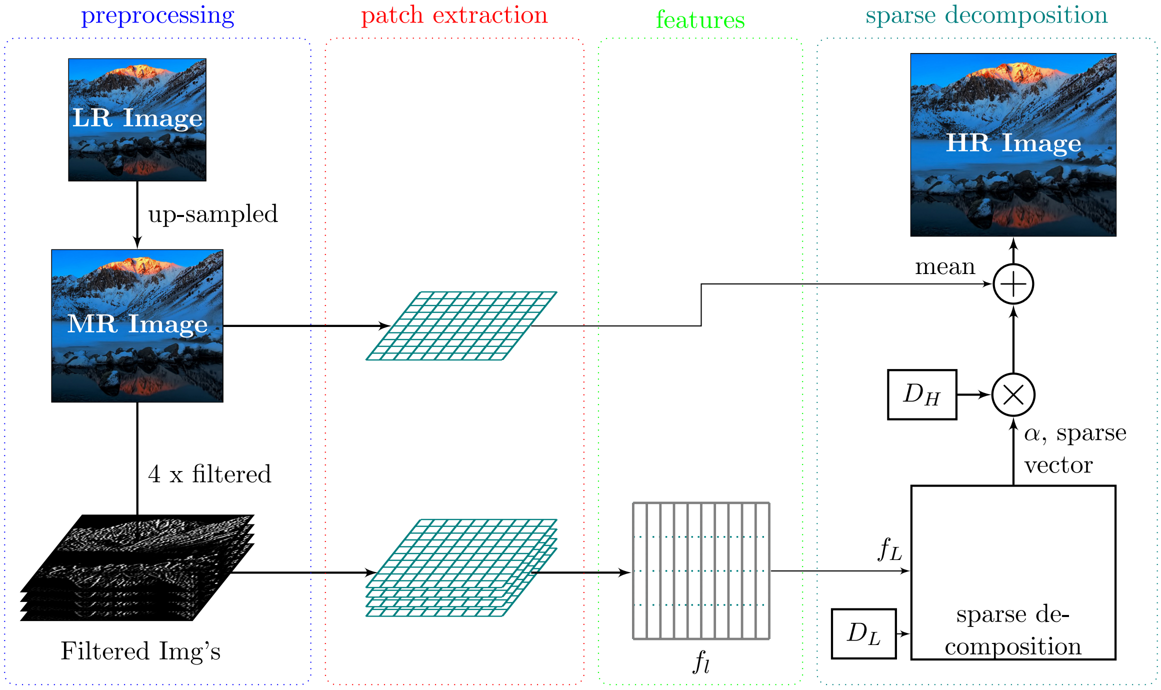


Figure 4.4.: Semantic overview of the SR test example preprocessing. The image preprocessing can be divided in four section, image preprocessing, patch extraction, feature concatenation and sparse decomposition. First the LR input image is bicubically up-sampled to the MR image. Next the MR image is filtered by applying the first and second order central differences, horizontally and vertically. Then we extract patches from the filtered mid-resolution images and co-instantaneously extract the mean of the unfiltered patches. The filtered patches are concatenated and form the LR feature used by the sparse decomposition. The resulting sparse vector is multiplied with the HR dictionary and the mean of the MR patch is added to form an estimated patch.

ner". Since the images of the test set "Mauthner" are already distorted, we can not compare them to the undistorted images but we compare the results to each other. For an upscaling factor of 2 we could not evaluate the method of Yang et al. and therefore left out because their dictionaries where corrupted and we did not want to give false results.

In order to evaluate our algorithms, sparse decomposition was done with FISTA for the active set algorithm 4 and IPIASCO for the smoothed $l_{1,\epsilon}$ regularized algorithm 3. Since no real performance tweaks were employed the algorithms have rather slow run-times which we note, as an average for all performed test cases, in the tables. To overcome this drawback we used our trained dictionaries in combination with an optimized fast solver from the Sparse Modeling Software (SPAMS) toolbox[MBPS09] and could achieve big run-time improvements for slightly inferior results. These averaged results are also stated in the evaluation tables.

Parameters	Values
Dictionary atoms	1024
Scaling factor	2
High-res. patch size	6
Low-res. patch size	3
Mid-res. patch size	6
λ_{active}	0.10
$\lambda_{smoothed}$	0.03
Max nr. of iterations	500

Table 4.1.: This table shows the parameters for testing dataset "Set 14" and "Li He" with scaling factor of 2.

The SPAMS toolbox incorporates many sparse solvers including Least Absolute Shrinkage and Selection Operator (LASSO) with elastic-net regularization. This regularization is quite similar to the $l_{1,\epsilon}$ regularization since it combines l_1- and l_2-regularization. At this point we have to note that this is mathematically not consistent but acceptable for practical consideration.

4.4.1. Test Results for Upscaling Factor of 2

Table 4.1 shows the simulation parameters used with magnification factor of 2 for both test sets. As already mentioned in the previous section, the λ values differ a lot for the two algorithms, the active set method and the smoothed $l_{1,\epsilon}$ regularized method. This can be explained by the use of the smoothed $l_{1,\epsilon}$-norm where the ϵ parameter influences the regularization parameter λ and therefore λ needs to be lower to get the same number of non-zero coefficients in the sparse vector. We choose to have a mean of 10 non-zeros entries in the sparse vector for a single patch resulting in the presented parameters.

Table 4.2 shows the evaluation results on the test set "Set14" for a magnification factor of 2. We could not evaluate the results of Yang et al.[YWHM10] because the shipped dictionaries included errors and we did not want to give false results. Interestingly, our methods outperform the others in terms of PSNR but the method of Timofte et al. achieves slightly better results in terms of the SSIM-index due to the superior elaboration of textured regions.

Table 4.3 shows the evaluation results on the test set "Li He" for upscaling factor of 2. We see that for a majority of the images we can outperform all other methods in terms of PSNR and SSIM. The differences in regard to the SSIM-index are minor. Interestingly, for specific image content the method of Timofte

system:	Bicubic		Yang et al.	Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
baboon	24.86	0.955	-	25.47	0.984	25.54	**0.986**	25.53	0.978	**25.60**	0.986
barbara	28.00	0.963	-	**28.70**	0.985	28.59	**0.986**	28.49	0.975	28.49	0.984
bridge	26.58	0.974	-	27.55	0.991	27.54	0.993	27.63	0.989	**27.69**	**0.993**
coastguard	29.12	0.789	-	30.41	0.840	30.44	**0.845**	30.30	0.815	**30.58**	0.840
comic	26.02	0.849	-	27.65	0.899	27.77	0.902	28.11	0.907	**28.15**	**0.910**
face	34.83	0.862	-	35.57	0.882	**35.63**	**0.884**	34.97	0.855	35.55	0.879
flowers	30.37	0.899	-	32.28	0.927	32.29	0.929	32.61	0.922	**32.73**	**0.931**
foreman	34.14	0.952	-	36.18	0.967	36.40	**0.967**	36.26	0.957	**36.52**	0.967
lenna	34.70	0.990	-	36.21	0.996	**36.32**	**0.997**	35.88	0.985	36.30	0.994
man	29.25	0.981	-	30.44	0.994	30.47	**0.994**	30.58	0.985	**30.72**	0.993
monarch	32.94	0.995	-	35.75	0.999	35.71	**0.999**	36.33	0.995	**36.40**	0.997
pepper	34.97	0.993	-	36.59	0.997	36.39	**0.997**	36.29	0.986	**36.73**	0.995
ppt3	26.87	0.991	-	29.30	0.998	28.97	0.998	**29.92**	0.998	29.82	**0.998**
zebra	30.63	0.987	-	33.21	0.997	33.07	**0.997**	32.94	0.991	**33.31**	0.997
average	30.23	0.941	-	31.81	0.961	31.80	**0.962**	31.85	0.953	**32.04**	0.962
mean run-time [s]	-	-	358	22	-	**2**	-	2398	-	14234	-
average, LASSO	30.23	0.941	-	31.81	0.961	31.80	**0.962**	31.92	0.959	**31.94**	0.957
mean run-time, LASSO [s]	-	-	-	22	-	**2**	-	14.9	-	15.4	-

Table 4.2.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] on the test set “Set 14” from [TDG13] for a scaling factor of 2.

et al. achieves better results and especially for images of face and animals (group gnd2x and gnd4x) they can outperform both of our algorithms.

To investigate this fact we present two exemplar images for a magnification factor of 2. Figure 4.5 shows the estimates of the image “monarch” with qualitative results. We see that our smoothed $l_{1,\epsilon}$-regularized method can outperform all others. This method can reduce ringing artifacts at edges and corners while still inferring fine texture. Our active set method also reduces the ringing at edges compared to the others but results in overall smoother images. Figure 4.6 shows the results of the image “gnd48” upscaled by factor of 2. This image belongs to the group of animal images where the method of Timofte et al. gives superior results compared to ours. Their system better infers textual content present in this image group like hairs and fur. The active set methods smooths the image at textured regions more than others method.

Figure 4.7 shows the qualitative and objective measurements of the estimates compared to bicubic interpolation, while figure 4.8 shows the results aggregated in a dataplot. Interestingly, our active set method achieves good performance in terms of PSNR but can not compete with the others in terms of SSIM due to the high smoothing of textured regions.



(a) original (b) bicubic 32.94dB (c) Zeyde et al. 35.75dB



(d) original cut (e) bicubic cut (f) Zeyde cut



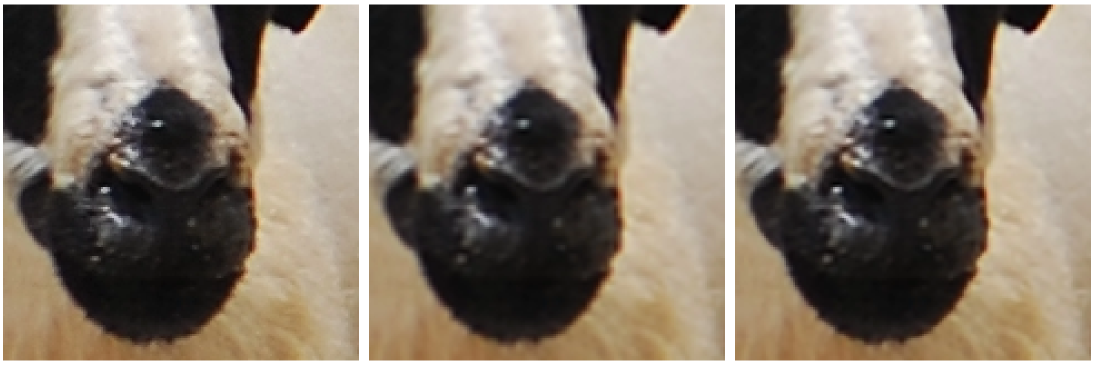
(g) Timofte et al. 35.71dB (h) active set 36.40dB (i) smoothed $l_{1,\epsilon}$ 32.94dB



(j) Timofte cut (k) active set cut (l) smoothed $l_{1,\epsilon}$ cut

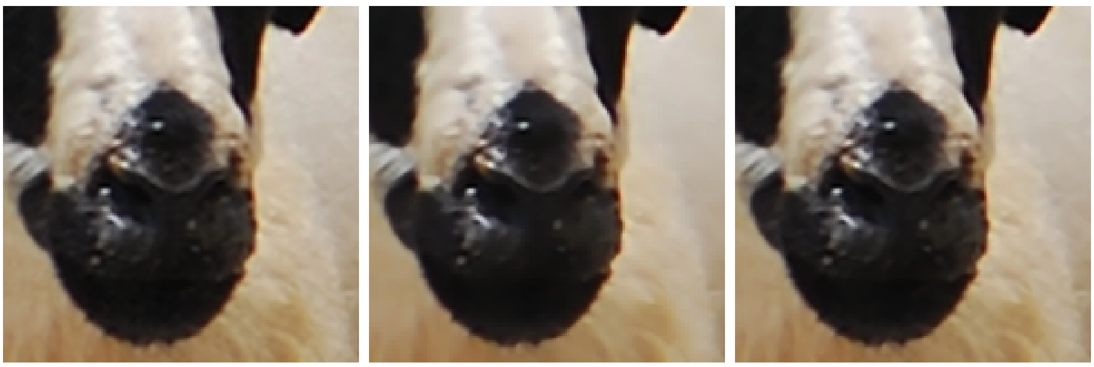
Figure 4.5.: High-resolution estimates of the monarch image upscaled by factor 2. Bicubic interpolation achieve a PSNR of 32.94dB, Zeyde et al. 35.75dB, Timofte et al. 35.71dB while our active set achieves 36.33dB and the smoothed $l_{1,\epsilon}$ regularized method **36.40dB**. This exemplar shows that our methods reduce ringing artifacts at edges and corners compared to the others.

(a) original (b) bicubic 37.55dB (c) Zeyde et al. 41.26dB



(d) original cut (e) bicubic cut (f) Zeyde cut

(g) Timofte et al. 41.34dB (h) active set 39.58dB (i) smoothed $l_{1,\epsilon}$ 40.48dB

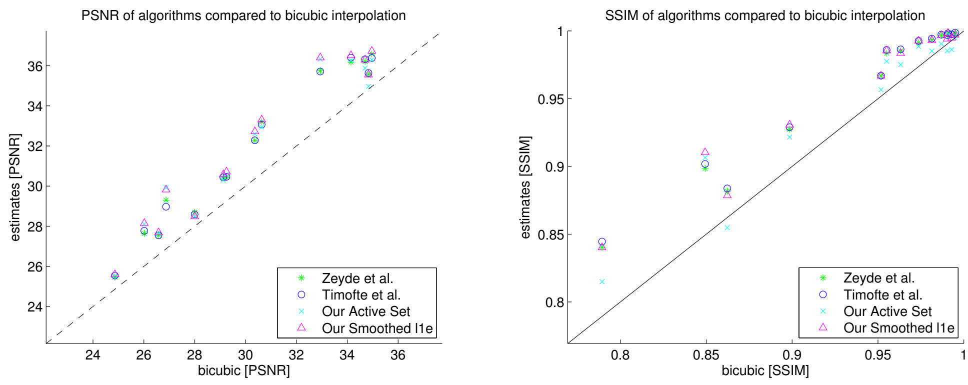


(j) Timofte cut (k) active set cut (l) smoothed $l_{1,\epsilon}$ cut

Figure 4.6.: High-resolution estimates of the gnd48 image upscaled by factor 2. Bicubic interpolation achieve a PSNR of 37.55dB, Zeyde et al. 41.26dB, Timofte et al. **41.34dB** while our active set achieves 39.58dB and the smoothed $l_{1,\epsilon}$ regularized method 40.48dB. This exemplar shows that Timofte et al. can infere more textured details compared to our methods.

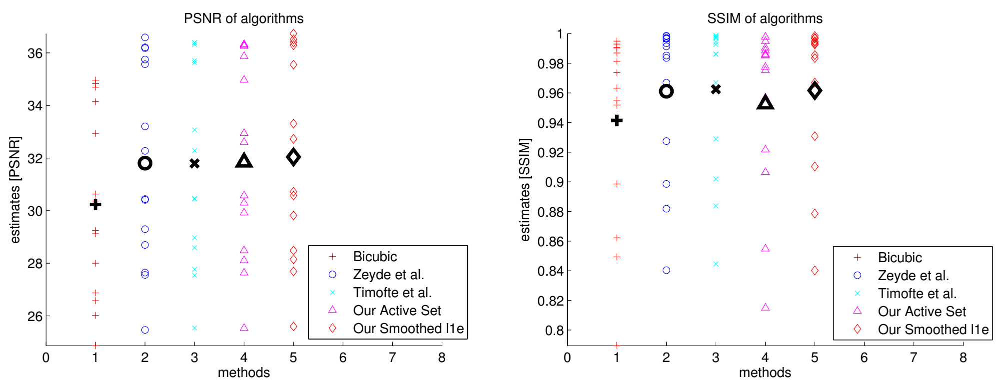
system:	Bicubic		Yang et al.	Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR -	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
gnd02	28.72	0.887	-	30.52	0.924	30.76	0.927	31.52	0.926	**31.56**	**0.935**
gnd03	28.65	0.831	-	29.79	0.873	29.85	0.876	29.94	0.865	**30.05**	**0.879**
gnd04	27.19	0.866	-	28.35	0.900	28.36	0.901	28.57	0.898	**28.64**	**0.906**
gnd05	27.35	0.889	-	28.97	0.922	28.96	0.922	29.45	0.920	**29.49**	**0.929**
gnd06	29.98	0.850	-	31.14	0.884	31.16	**0.886**	31.16	0.872	**31.28**	0.886
gnd07	30.22	0.919	-	32.06	0.943	32.04	0.943	32.56	0.939	**32.63**	**0.947**
gnd08	28.51	0.896	-	29.70	0.923	29.77	0.925	29.89	0.916	**30.01**	**0.927**
gnd09	30.53	0.922	-	32.44	0.948	32.29	0.945	32.87	0.946	**32.91**	**0.953**
gnd10	27.68	0.856	-	29.58	0.898	29.41	0.894	30.07	0.897	**30.09**	**0.905**
gnd12	29.69	0.841	-	31.03	0.883	31.22	0.889	31.18	0.878	**31.34**	**0.892**
gnd13	26.58	0.806	-	27.62	0.849	27.67	0.853	27.66	0.842	**27.72**	**0.853**
gnd14	26.61	0.765	-	27.51	0.816	27.64	0.823	27.58	0.809	**27.70**	**0.823**
gnd15	29.42	0.834	-	30.36	0.872	**30.45**	**0.876**	30.29	0.861	30.43	0.875
gnd16	31.19	0.872	-	32.08	0.898	**32.11**	**0.900**	31.91	0.881	32.09	0.898
gnd17	26.59	0.788	-	27.65	0.842	27.78	0.849	27.81	0.833	**27.91**	**0.851**
gnd18	24.76	0.772	-	25.61	0.820	25.71	0.826	25.80	0.816	**25.82**	**0.828**
gnd19	28.35	0.811	-	29.48	0.857	29.66	0.863	29.65	0.851	**29.84**	**0.866**
gnd20	29.46	0.864	-	30.35	0.896	30.46	**0.900**	30.40	0.889	**30.49**	0.900
gnd21	32.71	0.910	-	34.49	0.934	34.68	0.936	34.71	0.924	**34.99**	**0.936**
gnd23	31.45	0.938	-	34.18	0.956	34.04	0.956	34.58	0.949	**34.97**	**0.958**
gnd24	31.65	0.938	-	33.62	0.957	33.90	0.959	34.04	0.950	**34.30**	**0.960**
gnd25	42.43	0.979	-	43.53	0.984	**43.81**	**0.984**	41.20	0.967	43.16	0.981
gnd26	33.13	0.944	-	34.96	0.961	35.07	**0.962**	34.91	0.954	**35.18**	0.962
gnd27	33.07	0.916	-	34.46	0.939	34.60	**0.941**	34.29	0.919	**34.72**	0.939
gnd28	41.78	0.974	-	43.67	0.982	**44.10**	**0.983**	41.52	0.968	42.60	0.980
gnd29	38.75	0.973	-	40.92	0.980	**40.99**	**0.981**	39.82	0.968	40.49	0.979
gnd30	23.01	0.669	-	23.49	0.724	23.56	0.732	23.55	0.719	**23.61**	**0.734**
gnd31	28.08	0.807	-	29.15	0.845	29.06	0.844	29.32	0.838	**29.36**	**0.848**
gnd33	31.62	0.928	-	33.59	0.955	33.46	0.953	**34.03**	0.956	34.02	**0.958**
gnd34	26.31	0.805	-	27.35	0.846	27.34	0.847	**27.59**	0.845	27.59	**0.852**
gnd35	30.32	0.896	-	31.83	0.928	31.69	0.926	31.92	0.926	**31.94**	**0.931**
gnd36	28.23	0.863	-	29.58	0.900	29.62	0.902	**29.91**	0.901	29.90	**0.907**
gnd37	26.34	0.849	-	27.75	0.891	27.51	0.886	**28.04**	0.895	27.98	**0.898**
gnd38	26.21	0.785	-	27.29	0.841	27.40	0.847	27.42	0.840	**27.48**	**0.850**
gnd39	21.25	0.776	-	22.67	0.841	22.64	0.841	**23.04**	0.852	22.97	**0.852**
gnd40	25.82	0.816	-	27.42	0.865	27.41	0.866	27.80	0.863	**27.82**	**0.873**
gnd41	30.95	0.853	-	32.21	0.893	32.45	**0.898**	32.24	0.876	**32.53**	0.896
gnd42	32.83	0.880	-	33.61	0.902	**33.70**	**0.905**	33.33	0.888	33.57	0.902
gnd43	26.07	0.705	-	26.56	0.754	26.62	0.761	26.54	0.743	**26.63**	**0.761**
gnd45	36.73	0.969	-	38.94	0.978	**39.15**	**0.979**	38.52	0.967	39.12	0.977
gnd46	33.35	0.932	-	34.91	0.952	**35.17**	**0.955**	34.46	0.941	34.90	0.952
gnd47	35.80	0.944	-	37.78	0.962	37.83	**0.963**	37.40	0.952	**37.92**	0.962
gnd48	37.55	0.976	-	41.26	0.985	**41.34**	**0.986**	39.58	0.970	40.48	0.983
gnd49	27.77	0.808	-	28.90	0.852	28.97	**0.856**	28.89	0.839	**29.03**	0.856
gnd50	29.81	0.836	-	30.76	0.876	**30.89**	**0.881**	30.65	0.863	30.87	0.879
gnd52	25.86	0.781	-	26.77	0.833	26.89	0.839	26.82	0.831	**26.92**	**0.841**
gnd53	34.13	0.935	-	36.22	0.959	36.34	0.960	36.42	0.955	**36.68**	**0.964**
gnd54	37.43	0.950	-	39.53	0.967	39.65	**0.968**	38.79	0.952	**39.84**	0.968
gnd55	29.97	0.875	-	31.24	0.912	31.46	0.918	31.43	0.912	**31.59**	**0.920**
gnd56	28.72	0.878	-	30.03	0.913	30.14	0.916	30.16	0.908	**30.26**	**0.917**
gnd57	25.04	0.829	-	26.44	0.877	26.53	0.881	26.74	0.881	**26.76**	**0.886**
gnd58	28.72	0.878	-	30.33	0.914	30.36	0.917	30.45	0.910	**30.58**	**0.919**
gnd59	32.24	0.931	-	34.20	0.956	34.28	0.957	34.26	0.948	**34.52**	**0.958**
gnd60	27.98	0.898	-	29.76	0.932	29.92	0.935	30.22	0.935	**30.30**	**0.940**
gnd61	24.34	0.831	-	25.75	0.872	25.83	0.874	**26.32**	0.882	26.31	**0.884**
gnd63	32.34	0.960	-	36.43	0.977	36.21	0.978	37.04	0.972	**37.62**	**0.979**
gnd64	30.82	0.891	-	32.22	0.915	32.38	**0.917**	32.19	0.906	**32.46**	0.916
gnd65	26.20	0.872	-	28.11	0.913	27.98	0.914	29.04	0.917	**29.13**	**0.923**
gnd66	31.73	0.939	-	34.57	0.963	34.77	0.964	34.60	0.956	**35.13**	**0.965**
gnd67	29.16	0.891	-	31.50	0.927	31.53	0.928	32.05	0.922	**32.29**	**0.934**
gnd68	26.55	0.897	-	28.61	0.933	28.78	0.936	29.16	0.924	**29.37**	**0.938**
gnd69	26.90	0.868	-	28.88	0.906	28.92	0.908	29.50	0.912	**29.58**	**0.917**
gnd70	24.77	0.852	-	27.08	0.899	27.11	0.901	27.65	0.897	**27.69**	**0.905**
gnd71	27.86	0.876	-	29.52	0.913	29.61	0.915	29.94	0.907	**30.02**	**0.920**
gnd72	26.77	0.867	-	28.49	0.912	28.62	0.915	**29.19**	0.922	29.17	**0.924**
gnd73	28.89	0.917	-	30.61	0.941	30.60	0.942	31.02	0.943	**31.09**	**0.946**
gnd74	27.11	0.852	-	28.35	0.891	28.45	0.895	28.44	0.885	**28.53**	**0.896**
gnd75	29.91	0.905	-	31.87	0.938	31.97	0.941	32.55	0.934	**32.85**	**0.945**
gnd76	31.34	0.908	-	33.01	0.939	33.21	0.942	33.24	0.939	**33.47**	**0.946**
gnd77	24.53	0.715	-	25.27	0.775	25.35	0.782	25.32	0.773	**25.41**	**0.786**
gnd79	26.61	0.778	-	27.61	0.826	27.66	0.829	27.76	0.820	**27.80**	**0.833**
gnd80	30.33	0.884	-	31.90	0.920	32.03	0.923	32.34	0.918	**32.44**	**0.928**
average	29.59	0.869	-	31.16	0.904	31.23	0.906	31.25	0.898	**31.47**	**0.909**
mean run-time [s]	-	-	76.9	2.9	-	**0.4**	-	345.6	-	769.8	-
average, LASSO	29.59	0.869	-	31.16	0.904	31.23	0.906	**31.35**	**0.907**	31.33	0.902
mean run-time, LASSO [s]	-	-	-	4.8	-	**0.4**	-	5.2	-	5.3	-

Table 4.3.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12], Timofte et al.[TDG13] on the test set “Li He” from [HQZ13] for a scaling factor of 2.



(a) psnr (b) ssim

Figure 4.7.: This figures show the performance of the different methods compared to bicubic interpolation upscaled by a factor of 2. We can see that our $l_{1,\epsilon}$-regularized method achieves better performance in regard of PSNR while the method of Timofte et al. outperforms the others in terms of the SSIM-index.

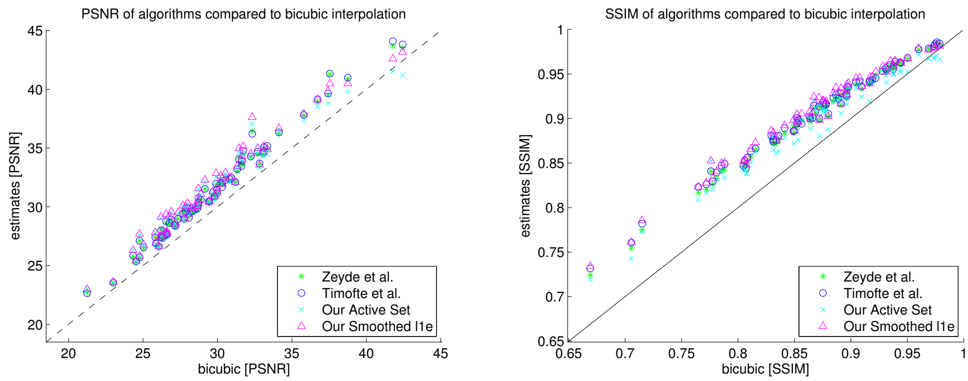


(a) psnr (b) ssim

Figure 4.8.: This figure shows the aggregated results for magnification factor of 2 on the test set "Set14". The bold markers represent the average of the results. Interesting to see is that our active set method achieves good performance in terms of PSNR but can not compete with the others in terms of the SSIM-index.

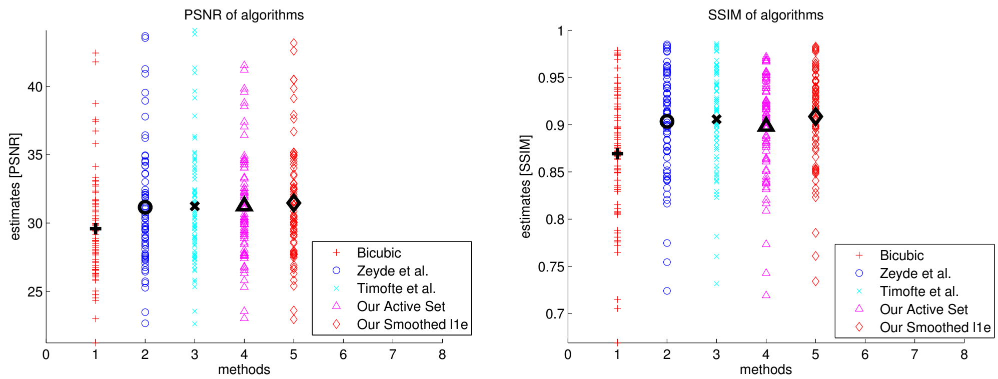
4.4.2. Test Results for Upscaling Factor of 3

Table 4.4 shows the simulation parameters used with magnification factor of 3 for both test sets. Again we choose to have a mean of 10 non-zeros entries in the sparse vector for a single patch resulting in the presented parameters where λ_{active} is set to 0.1 while $\lambda_{smoothed}$ is set to be 0.03 to reach the same number of non-zero entries.



(a) psnr (b) ssim

Figure 4.9.: This figures show the performance of the different methods compared to bicubic interpolation upscaled by a factor of 2 on the test set "LI He". We can see that our $l_{1,\epsilon}$-regularized method can outperform all the others for most of the images.



(a) psnr (b) ssim

Figure 4.10.: This figure shows the aggregated results for magnification factor of 2 on the test set "Li He". We can see that our $l_{1,\epsilon}$-regularized method achieves best performance in regard of PSNR and SSIM-index, while the active set method suffers specially in terms of the SSIM index.

Parameters	Values
Dictionary atoms	1024
Scaling factor	3
High-res. patch size	9
Low-res. patch size	3
Mid-res. patch size	6
λ_{active}	0.10
$\lambda_{smoothed}$	0.03
ϵ	10^{-5}
Max nr. of iterations	500

Table 4.4.: This table shows the parameters for testing dataset "Set 14" and "Li He" with scaling factor of 3.

system:	Bicubic		Yang et al.		Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
baboon	23.21	0.805	23.46	0.843	23.52	0.846	23.56	**0.851**	23.49	0.828	**23.58**	0.849
barbara	26.25	0.877	26.39	0.884	**26.77**	0.899	26.70	**0.899**	26.56	0.883	26.75	0.897
bridge	24.40	0.865	24.78	0.896	25.02	0.899	25.00	0.902	24.98	0.888	**25.11**	**0.902**
coastguard	26.55	0.615	26.95	0.638	27.14	0.655	27.07	**0.658**	27.02	0.616	**27.20**	0.648
comic	23.12	0.699	23.84	0.754	23.98	0.756	24.01	0.759	24.19	0.762	**24.29**	**0.773**
face	32.82	0.798	33.07	0.801	33.53	0.820	**33.60**	**0.823**	33.00	0.789	33.57	0.816
flowers	27.23	0.801	28.22	0.829	28.41	0.837	28.44	0.839	28.58	0.829	**28.80**	**0.845**
foreman	31.18	0.906	32.22	0.911	33.15	0.929	33.16	0.929	33.27	0.920	**33.77**	**0.933**
lenna	31.68	0.953	32.43	0.956	32.99	0.967	33.07	**0.968**	32.89	0.953	**33.27**	0.965
man	27.01	0.909	27.70	0.926	27.90	0.934	27.91	**0.936**	27.96	0.919	**28.17**	0.936
monarch	29.43	0.970	30.63	0.976	31.09	0.981	31.02	0.981	31.50	0.978	**31.83**	**0.982**
pepper	32.39	0.969	33.23	0.964	34.02	**0.978**	33.76	0.978	34.00	0.965	**34.44**	0.976
ppt3	23.71	0.942	24.88	0.960	25.22	0.965	24.96	0.962	25.60	0.968	**25.74**	**0.971**
zebra	26.63	0.912	27.81	0.933	28.51	0.941	28.40	**0.942**	28.51	0.921	**28.93**	0.941
average	27.54	0.859	28.26	0.876	28.66	0.886	28.62	0.888	28.68	0.873	**28.96**	**0.888**
mean run-time [s]	-	-	92.9	-	3.7	-	**0.7**	-	486.2	-	1156.5	-
average, LASSO	27.54	0.859	27.54	0.859	28.66	0.886	28.62	**0.888**	28.74	0.884	**28.87**	0.879
mean run-time [s], LASSO	-	-	92.9	-	6.4	-	**0.7**	-	8.5	-	8.5	-

Table 4.5.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] on the test set “Set 14” from [TDG13] for a scaling factor of 3. We see that both our algorithms gain performance compare to the results upscaled by factor 2.

Table 4.5 shows the evaluation results on the test set “Set14” for a magnification factor of 3. For this upscaling factor we could evaluate the results of Yang et al.[YWHM10] due to the correctness of the shipped dictionaries. Interestingly, for this upscaling factor our $l_{1,\epsilon}$-regularized method outperform the others in terms of both measurements, the PSNR and the SSIM-index.

Table 4.6 shows the evaluation results on the test set “Li He” for an upscaling factor of 3. Compared to the result of magnification factor 2 both our algorithms gain performance in terms of PSNR and SSIM. We explain this fact by the use of bilevel optimization. Our bilevel programs are able to train the dictionaries such that they are optimal in both feature spaces individually. We think that this capacity is beneficial and has more impact for higher scaling factors.

We present three exemplar images for this magnification factor. Figure 4.11 shows the estimates of the image “zebra” with qualitative results. We can see that the smoothed $l_{1,\epsilon}$ method reduces the ringing artifacts at the leg of the zebra compared to others. Figure 4.12 shows the results of the image “gnd63” upscaled by factor of 3. This is a rare case where the active set method outperforms all the others. This phenomena can be explained by the content of the image. Computer tomography images mainly consist of flat regions separated by strong edges. Since this is also a characteristic result for the active set method, it performs best on this image group. At last figure 4.13 shows the results of the image “gnd48”. Here we see that the method of Timofte et al. can infer more details at textured regions, for example for the hairs of the girl. Our smoothed $l_{1,\epsilon}$ method is competitive for this group of images but does not outperform Timofte et al. for this image.

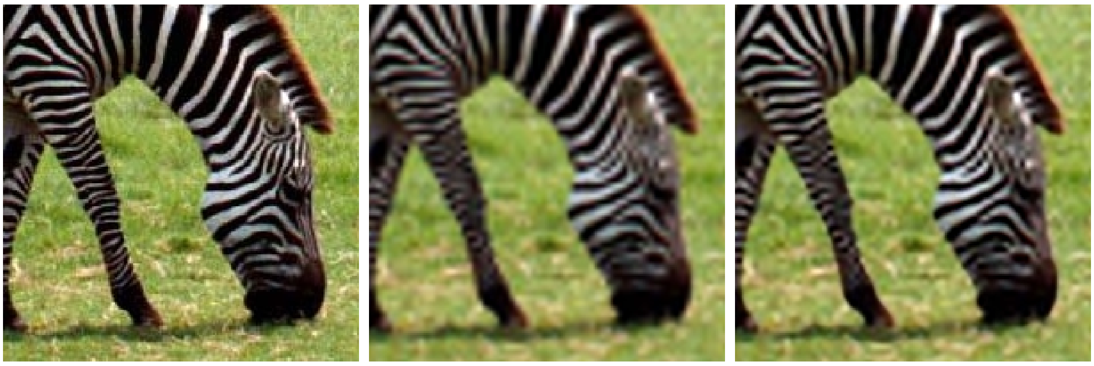
Figure 4.14 shows the results of upscaling factor 3 on the test set "Li He" compared to bicubic interpolation. We see that our $l_{1,\epsilon}$-regularized method achieves best overall performance in regard of PSNR and the SSIM-index, while Timofte et al. are better when bicubic interpolation performs well. We explain this by their training scheme. Since Timofte et al. only learn the differences between bicubic interpolation and the actual HR patch they "start" already from a higher level before inferring novel details. Figure 4.15 shows the aggregated results in terms of PSNR and SSIM-index for magnification factor 3 on the test set "Li He". We can see that our $l_{1,\epsilon}$-regularized method achieves best overall performance in regard of PSNR and the SSIM-index.

4.4.3. Test Results for Degenerated Images

In order to investigate the performance of SR systems on degenerated images we took some images of a real-world example. These images where take automatically on a skiing slope and on a car test track. Since these images are already distorted and no ground truth is available, we can only compare the results subjectively to each other.

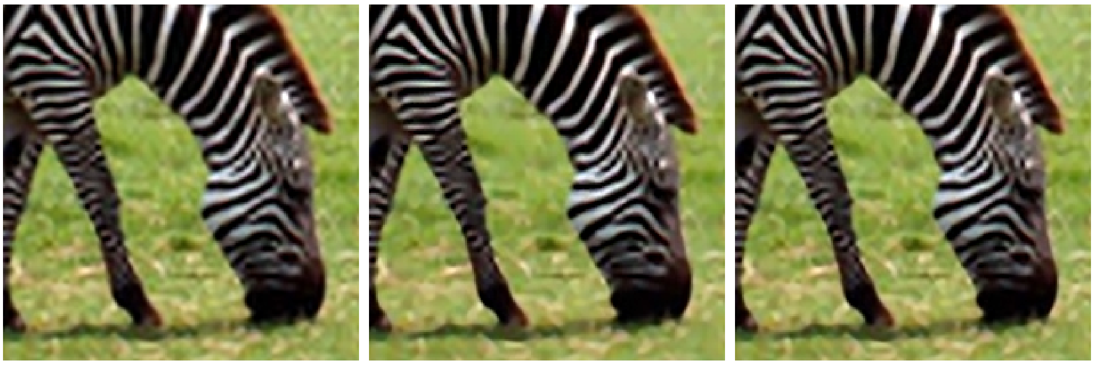
In table 4.7 we give qualitative measurements on the noisy "Set14". We added zero-mean white Gaussian noise with a standard deviation of 0.01 to the images which have been in the rage between [0..1]. Due to the higher smoothing of the images, the active set method performs best. This can also be seen in figure 4.16 where we present the results of the noisy image "coastguard". In figure 4.17 we present the results of the real-world example "BMW02". Since this image set is already degenerated, no qualitative evaluation was performed. We see that all algorithms yield more or less equal results but subjectively our methods seem slightly better for example at the road paintings and car boarders.

(a) original (b) Yang et al. 27.81dB (c) Zeyde et al. 28.51dB



(d) original cut (e) Yang cut (f) Zeyde cut

(g) Timofte et al. 28.40dB (h) active set 28.51dB (i) smoothed $l_{1,\epsilon}$ 28.93dB

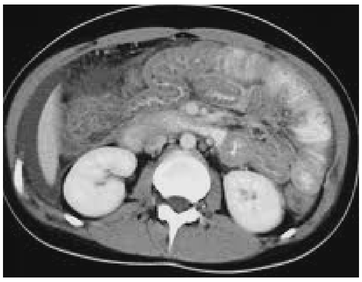


(j) Timofte cut (k) active set cut (l) smoothed $l_{1,\epsilon}$ cut

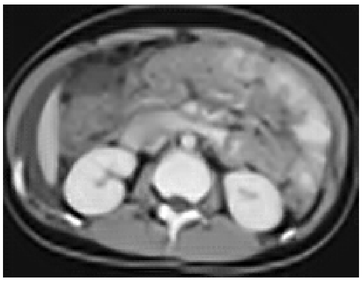
Figure 4.11.: High-resolution estimates of the zebra image upscaled by factor 3. Yang et al. achieve a PSNR of 27.81dB, Zeyde et al. 28.51dB, Timofte et al. 28.40dB while our active set achieves 28.51dB and the smoothed $l_{1,\epsilon}$ regularized method **28.93dB**. We can see that the smoothed $l_{1,\epsilon}$ method reduces the ringing artifacts compared to others.

system:	Bicubic		Yang et al.		Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
gnd02	26.23	0.796	27.17	0.827	27.18	0.835	27.19	0.834	27.47	0.835	**27.67**	**0.850**
gnd03	26.50	0.723	27.03	0.754	27.25	0.764	27.23	0.766	27.27	0.751	**27.44**	**0.771**
gnd04	24.99	0.763	25.50	0.792	25.67	0.799	25.66	0.797	25.69	0.794	**25.77**	**0.804**
gnd05	24.51	0.787	25.09	0.813	25.48	0.825	25.41	0.823	25.61	0.821	**25.72**	**0.834**
gnd06	27.46	0.739	27.93	0.766	28.21	0.776	28.17	0.777	28.12	0.757	**28.33**	**0.778**
gnd07	27.32	0.842	28.30	0.859	28.45	0.870	28.37	0.868	28.63	0.863	**28.80**	**0.876**
gnd08	25.93	0.809	26.82	0.834	26.79	0.844	26.78	0.844	27.01	0.840	**27.22**	**0.854**
gnd09	27.41	0.839	28.08	0.862	28.53	0.875	28.39	0.869	**28.74**	0.875	28.73	**0.884**
gnd10	24.65	0.737	25.32	0.760	25.74	0.783	25.52	0.773	26.00	0.780	**26.11**	**0.796**
gnd12	27.01	0.706	27.46	0.743	27.81	0.752	27.89	0.759	27.84	0.739	**28.10**	**0.762**
gnd13	24.03	0.653	24.38	0.693	24.62	0.700	24.65	0.704	24.55	0.685	**24.72**	**0.704**
gnd14	24.37	0.603	24.79	0.649	24.93	0.655	24.99	**0.662**	24.88	0.635	**25.03**	0.659
gnd15	26.97	0.701	27.22	0.732	27.52	0.742	**27.57**	**0.748**	27.37	0.720	27.55	0.744
gnd16	29.09	0.789	29.13	0.793	29.67	0.815	29.68	**0.818**	29.44	0.790	**29.73**	0.814
gnd17	24.31	0.625	24.77	0.669	24.81	0.674	24.87	0.682	24.83	0.653	**24.96**	**0.682**
gnd18	22.78	0.632	23.06	0.657	23.17	0.670	23.18	0.673	23.21	0.658	**23.25**	**0.675**
gnd19	26.01	0.669	26.49	0.710	26.63	0.715	26.69	**0.721**	26.64	0.699	**26.79**	0.721
gnd20	27.38	0.768	27.70	0.790	27.85	0.799	27.89	0.802	27.78	0.784	**27.94**	**0.803**
gnd21	29.95	0.838	30.67	0.850	31.20	0.866	31.23	0.868	31.31	0.856	**31.64**	**0.872**
gnd23	28.28	0.883	29.68	0.894	30.24	0.912	30.09	0.910	30.73	0.906	**31.19**	**0.918**
gnd24	28.69	0.879	29.39	0.883	29.86	0.902	29.94	0.903	30.00	0.894	**30.29**	**0.907**
gnd25	39.03	0.957	39.17	0.956	41.20	0.972	**41.37**	**0.973**	38.56	0.943	40.52	0.967
gnd26	29.76	0.882	30.41	0.896	31.28	0.912	31.28	0.912	31.25	0.904	**31.60**	**0.915**
gnd27	30.38	0.842	30.74	0.849	31.19	0.866	31.27	**0.869**	31.00	0.840	**31.37**	0.865
gnd28	37.70	0.941	35.90	0.931	39.14	0.954	**39.40**	**0.956**	37.39	0.932	38.68	0.951
gnd29	35.67	0.951	34.60	0.938	**37.48**	0.961	37.45	**0.961**	36.37	0.944	37.27	0.959
gnd30	21.69	0.520	21.91	0.561	21.95	0.564	21.99	**0.572**	21.94	0.548	**22.01**	0.569
gnd31	25.95	0.692	26.44	0.715	26.66	0.727	26.55	0.724	26.73	0.717	**26.85**	**0.732**
gnd33	28.35	0.836	29.02	0.865	29.79	0.880	29.73	0.879	29.89	0.876	**30.02**	**0.886**
gnd34	24.35	0.694	24.67	0.709	24.84	0.726	24.82	0.726	24.87	0.716	**25.02**	**0.734**
gnd35	27.63	0.796	27.92	0.815	28.77	0.840	28.60	0.836	28.64	0.831	**28.84**	**0.844**
gnd36	25.61	0.746	26.15	0.780	26.44	0.789	26.43	0.790	26.54	0.787	**26.71**	**0.800**
gnd37	23.76	0.719	24.12	0.746	24.56	0.764	24.46	0.759	24.62	0.763	**24.67**	**0.771**
gnd38	23.60	0.595	24.05	0.654	24.12	0.653	24.16	0.661	24.12	0.641	**24.22**	**0.661**
gnd39	18.38	0.561	18.86	0.623	19.00	0.623	18.97	0.622	19.09	0.634	**19.13**	**0.639**
gnd40	23.10	0.669	23.93	0.712	24.13	0.724	24.07	0.723	24.35	0.724	**24.48**	**0.739**
gnd41	28.37	0.725	28.60	0.741	28.91	0.757	28.97	**0.761**	28.75	0.727	**29.02**	0.756
gnd42	30.73	0.804	30.35	0.804	31.24	0.826	**31.31**	**0.829**	30.87	0.802	31.16	0.823
gnd43	24.66	0.567	24.77	0.601	24.95	0.610	24.97	**0.617**	24.85	0.584	**24.98**	0.612
gnd45	32.71	0.932	33.44	0.928	34.08	0.945	34.28	**0.946**	33.88	0.930	**34.42**	0.945
gnd46	29.45	0.837	30.01	0.858	30.58	0.870	**30.67**	**0.873**	30.33	0.855	30.61	0.871
gnd47	32.28	0.877	32.91	0.889	33.60	0.903	33.56	0.904	33.34	0.889	**33.84**	**0.905**
gnd48	32.81	0.935	33.38	0.924	35.70	**0.954**	35.53	0.954	35.05	0.933	**35.93**	0.952
gnd49	25.32	0.675	25.67	0.699	26.01	0.716	26.02	**0.720**	25.95	0.695	**26.10**	0.719
gnd50	27.40	0.695	27.71	0.735	27.97	0.742	**28.04**	**0.750**	27.77	0.714	28.00	0.743
gnd52	23.42	0.600	23.77	0.662	23.91	0.658	**23.99**	**0.667**	23.90	0.649	23.98	0.666
gnd53	30.95	0.864	31.51	0.885	32.11	0.895	32.21	0.898	32.03	0.888	**32.45**	**0.904**
gnd54	34.14	0.895	34.74	0.904	35.52	0.918	35.57	0.920	34.92	0.897	**35.83**	**0.920**
gnd55	27.12	0.742	27.63	0.790	27.91	0.792	28.03	**0.800**	27.89	0.783	**28.05**	0.799
gnd56	25.80	0.752	26.38	0.791	26.51	0.795	26.62	0.800	26.53	0.784	**26.70**	**0.802**
gnd57	22.39	0.677	23.06	0.726	23.22	0.729	23.25	0.733	23.33	0.729	**23.42**	**0.741**
gnd58	25.84	0.759	26.64	0.800	27.08	0.808	27.09	0.813	27.16	0.802	**27.39**	**0.819**
gnd59	28.80	0.841	29.57	0.865	29.99	0.878	30.00	0.880	29.98	0.867	**30.24**	**0.884**
gnd60	24.62	0.772	25.40	0.814	25.61	0.818	25.71	0.823	25.81	0.821	**25.92**	**0.831**
gnd61	21.76	0.713	22.69	0.758	22.71	0.761	22.73	0.762	23.00	0.776	**23.07**	**0.781**
gnd63	28.01	0.896	29.98	0.915	30.47	0.928	30.44	0.928	31.19	0.924	**31.70**	**0.936**
gnd64	27.84	0.807	28.79	0.831	29.17	0.840	29.28	0.844	29.23	0.832	**29.53**	**0.845**
gnd65	23.34	0.739	24.49	0.788	24.39	0.789	24.30	0.788	**24.86**	0.798	24.80	**0.806**
gnd66	27.50	0.853	28.97	0.878	29.31	0.890	29.47	0.892	29.53	0.887	**29.90**	**0.898**
gnd67	25.53	0.776	27.08	0.816	26.98	0.822	27.17	0.826	27.38	0.820	**27.62**	**0.835**
gnd68	23.28	0.771	24.38	0.804	24.40	0.813	24.61	0.819	24.81	0.806	**25.08**	**0.825**
gnd69	23.78	0.758	25.16	0.809	25.18	0.810	25.27	0.813	25.63	0.822	**25.76**	**0.828**
gnd70	21.53	0.703	22.81	0.753	22.91	0.761	22.88	0.762	23.26	0.761	**23.36**	**0.775**
gnd71	24.97	0.759	25.66	0.785	25.88	0.799	25.87	0.801	26.00	0.786	**26.17**	**0.807**
gnd72	23.80	0.726	24.70	0.785	24.75	0.781	24.83	0.785	24.96	0.785	**25.07**	**0.798**
gnd73	25.93	0.840	26.58	0.862	26.77	0.866	26.74	0.867	**26.91**	0.868	26.90	**0.873**
gnd74	24.29	0.716	24.70	0.750	24.96	0.759	25.03	0.764	24.94	0.747	**25.08**	**0.766**
gnd75	26.59	0.790	27.23	0.818	27.69	0.832	27.70	0.834	27.77	0.823	**28.01**	**0.841**
gnd76	28.14	0.801	28.45	0.829	29.10	0.840	29.21	0.845	29.10	0.835	**29.38**	**0.852**
gnd77	22.62	0.538	22.97	0.598	23.04	0.595	23.08	0.604	23.05	0.586	**23.13**	**0.606**
gnd79	24.31	0.625	24.68	0.660	24.91	0.671	24.91	0.674	24.93	0.657	**25.06**	**0.677**
gnd80	27.68	0.781	28.43	0.815	28.67	0.824	28.70	0.826	28.78	0.815	**29.02**	**0.835**
average	26.76	0.760	27.32	0.788	27.75	0.799	27.78	0.801	27.72	0.789	**27.99**	**0.806**
mean run-time [s]	-	-	36.2	-	2.4	-	**0.5**	-	1353.8	-	1151.0	-
average, LASSO	26.76	0.760	26.76	0.760	27.75	0.799	27.78	0.801	27.81	**0.802**	**27.88**	0.795
mean run-time [s], LASSO	-	-	-	-	2.2	-	**0.3**	-	3.3	-	3.3	-

Table 4.6.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] on the test set “Li He” from [HQZ13] for a scaling factor of 3 .



(a) original



(b) Yang et al. 29.98dB

(c) Zeyde et al. 30.47dB

(d) Timofte et al. 30.44dB

(e) active set 31.19dB

(f) smoothed $l_{1,\epsilon}$ 31.70B

Figure 4.12.: High-resolution estimates of the gnd65 image upscaled by factor 3. Yang et al. achieve a PSNR of 29.98dB, Zeyde et al. 30.47dB, Timofte et al. 30.44dB while our active set achieves 31.19dB and the smoothed $l_{1,\epsilon}$ regularized method **31.70B**. For this class of images our methods outperform all others and reduce ringing artifacts.

system:	Bicubic		Yang et al.		Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
baboon	20.05	0.732	22.62	0.745	22.73	0.742	22.68	0.739	**22.83**	**0.759**	22.71	0.745
barbara	20.06	0.648	25.05	0.751	25.23	0.742	25.06	0.731	**25.45**	**0.789**	25.18	0.747
bridge	20.17	0.790	23.64	0.817	23.92	0.820	23.81	0.817	**23.98**	**0.828**	23.88	0.821
coastguard	20.05	0.350	25.40	0.543	25.48	0.541	25.31	0.533	**25.87**	**0.561**	25.50	0.541
comic	20.08	0.535	22.88	0.678	23.10	0.678	23.06	0.675	**23.25**	**0.697**	23.19	0.686
face	20.40	0.235	28.91	0.653	28.79	0.639	28.51	0.624	**29.76**	**0.694**	28.82	0.644
flowers	20.16	0.360	26.16	0.695	26.35	0.692	26.19	0.681	**26.74**	**0.732**	26.41	0.697
foreman	20.22	0.245	28.50	0.728	28.67	0.714	28.39	0.695	**29.59**	**0.790**	28.73	0.722
lenna	20.01	0.567	28.54	0.777	28.52	0.758	28.26	0.745	**29.48**	**0.828**	28.59	0.767
man	20.31	0.686	25.90	0.808	26.04	0.801	25.90	0.793	**26.41**	**0.836**	26.10	0.807
monarch	20.02	0.571	27.63	0.784	27.73	0.763	27.46	0.748	**28.62**	**0.842**	27.95	0.773
pepper	20.08	0.561	28.92	0.784	28.90	0.766	28.52	0.751	**30.00**	**0.839**	28.97	0.775
ppt3	20.86	0.647	23.96	0.817	24.14	0.790	23.88	0.776	**24.69**	**0.873**	24.48	0.807
zebra	20.19	0.754	25.93	0.840	26.41	0.844	26.18	0.838	**26.69**	**0.856**	26.50	0.844
average	20.19	0.549	26.00	0.744	26.14	0.735	25.94	0.725	**26.67**	**0.780**	26.22	0.741
mean run-time [s]	-	-	180.9	-	8.8	-	**0.9**	-	690.1	-	1353.7	-

Table 4.7.: This table shows the evaluation of our algorithms on the degenerate test set “Set14” for a scaling factor of 2. We see that our active set method works best for noisy data.

4.5. Discussion

We evaluated our two algorithms on two distinct dataset and two upscaling factors and presented the results. With all this data at hand we can make some basic assumptions regarding the tested algorithms. In general the dictionaries trained with the $l_{1,\epsilon}$ regularized bilevel program gave superior results for most test cases.

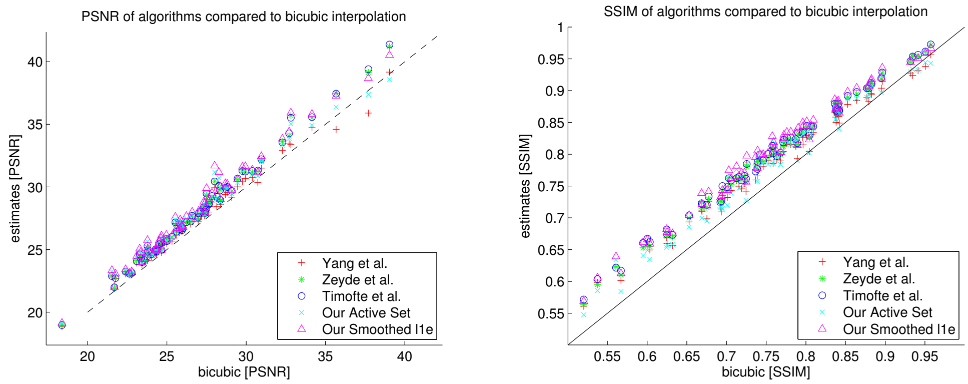
(a) original (b) Yang et al. 35.90dB (c) Zeyde et al. 39.14dB

(d) original cut (e) Yang cut (f) Zeyde cut

(g) Timofte et al. 39.40dB (h) active set 37.39dB (i) smoothed $l_{1,\epsilon}$ 38.68dB

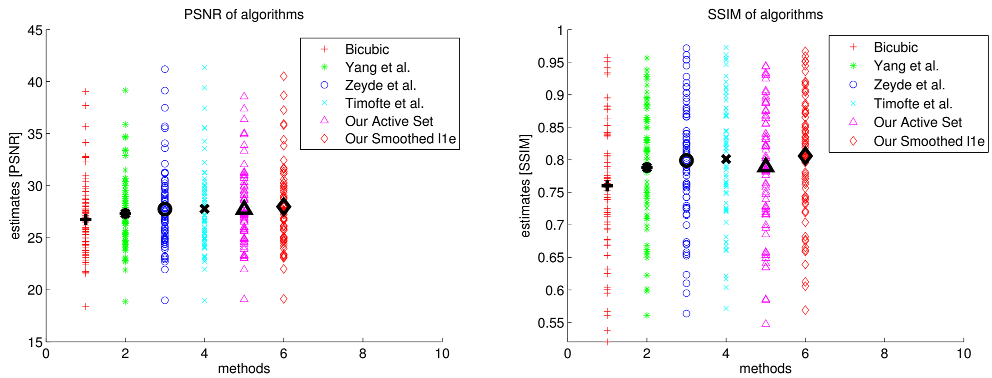
(j) Timofte cut (k) active set cut (l) smoothed $l_{1,\epsilon}$ cut

Figure 4.13.: High-resolution estimates of the gnd28 image upscaled by factor 3. Yang et al. achieve a PSNR of 35.90dB, Zeyde et al. 39.14dB, Timofte et al. **39.40dB** while our active set achieves 37.39dB and the smoothed $l_{1,\epsilon}$ regularized method 38.68dB. We see that the method of Timofte et al. can infer more details at textured regions for example the hairs of the girl.



(a) psnr (b) ssim

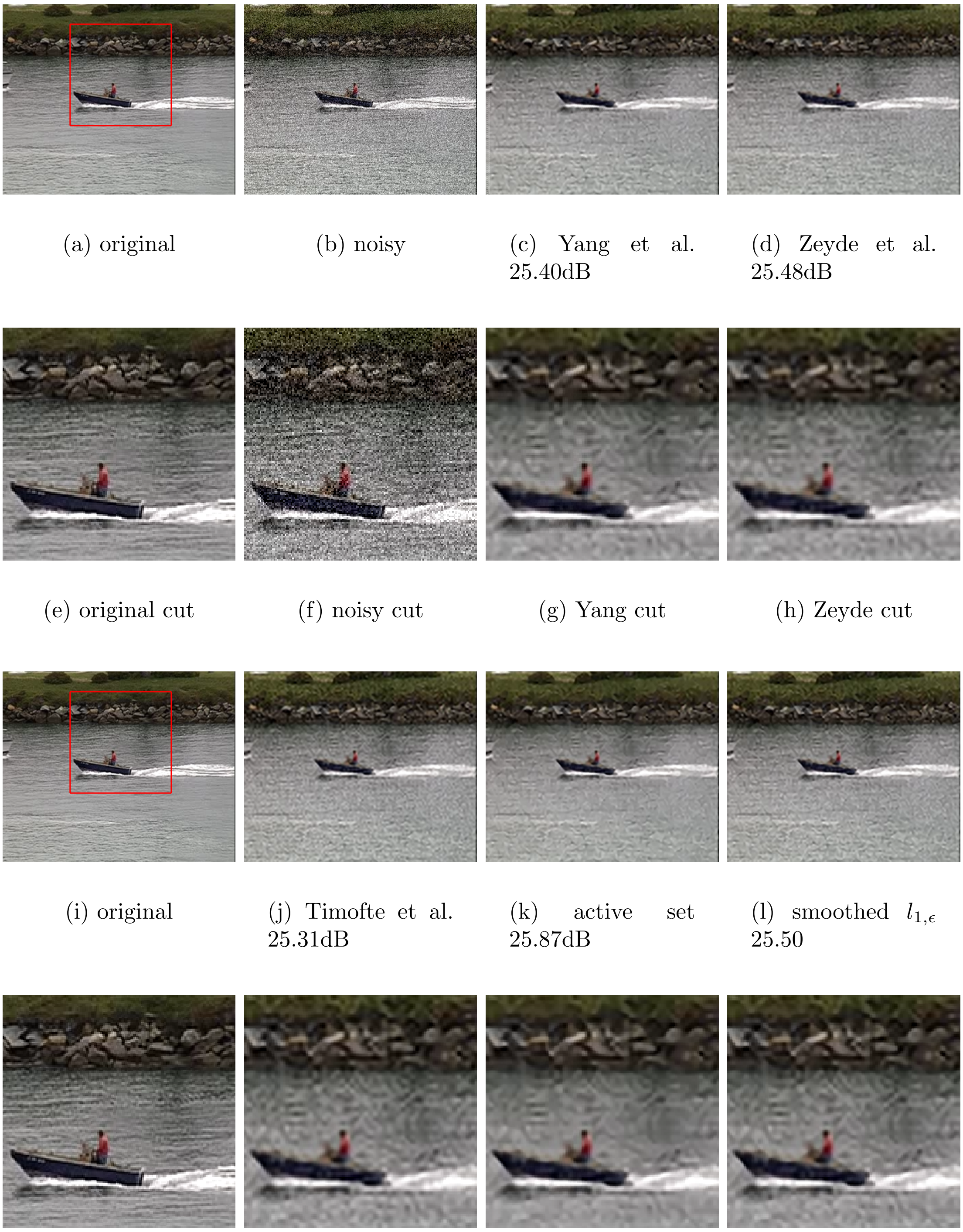
Figure 4.14.: This figures show the performance of the different methods compared to bicubic interpolation upscaled by a factor of 3. We can see that our $l_{1,\epsilon}$-regularized method achieves best overall performance in regard of PSNR and the SSIM-index, while Timofte et al. are better when bicubic interpolation performs well.



(a) psnr (b) ssim

Figure 4.15.: This figure shows the aggregated results for magnification factor of 3 on the test set "Li He". We can see that our $l_{1,\epsilon}$-regularized method achieves best overall performance in regard of PSNR and the SSIM-index.

This algorithm is capable of inferring fine structured details while reducing ringing and jaggies artifacts. This comes with the price of a rather slow run-time, although there is still a lot of improvement possible. The active set bilevel program trains the dictionaries such that they give overall smooth estimates with sharp edges and also reduces ringing and jaggies artifacts. It seems to give equal results as system solving the inverse problem formulation with Total Variation (TV) regularization. But in comparison to Timofte et al. or the $l_{1,\epsilon}$-program it is not capable to infer fine



(a) original (b) noisy (c) Yang et al. 25.40dB (d) Zeyde et al. 25.48dB

(e) original cut (f) noisy cut (g) Yang cut (h) Zeyde cut

(i) original (j) Timofte et al. 25.31dB (k) active set 25.87dB (l) smoothed $l_{1,\epsilon}$ 25.50

(m) original cut (n) Timofte cut (o) active set cut (p) smoothed $l_{1,\epsilon}$ cut

Figure 4.16.: High-resolution estimates of the noisy coastguard image upscaled by factor 3. Yang et al. achieve a PSNR of 25.40dB, Zeyde et al. 25.48dB, Timofte et al. 25.31dB while our active set achieves **25.87dB** and the smoothed $l_{1,\epsilon}$ regularized method 25.50. Due to the high smoothing of the active set method, their results perform best for noisy images.

(a) original (b) Yang et al. (c) Zeyde et al.

(d) Timofte et al. (e) active set (f) smoothed $l_{1,\epsilon}$

Figure 4.17.: High-resolution estimates of the distorted BMW02 image upscaled by factor 3. These images are already degenerated and therefor no qualitative evaluation was performed. We see that all algorithms yield equal results but subjectively our methods seem slightly better than the others.

details which we see in the evaluated SSIM-index. In general our two algorithms perform better for higher upscaling factors, namely the magnification factor 3. We think the reason is our bilevel training scheme since the dictionaries are trained such that they are optimal in both feature spaces individually and this fact is more emphasized at higher scaling factors. Due to the comprehensive dataset of “Li He” we could experience that some algorithms perform better for specific classes of images. For example Timofte et al. perform better on images with faces or animals where they can infer fine details. This class of images consists of many textured regions including hair and fur. For other classes like medical images the active set method proved to give good results. Since this type of images mainly consist of flat regions separated by sharp edges the active set method performs well. For degenerated noisy images the active set trained dictionaries can outperform all the others. As it does more smoothing then the others the noise in the images gets suppressed rather then augmented. We can say that this methods is more robust than others regarding the noise and would be the algorithm of choice for noisy data. For the general class of natural images the $l_{1,\epsilon}$ regularized program would be our choice since it can outperform the others in regard of PSNR and SSIM-index especially if higher upscaling factor are needed.

The main drawback our our bilevel program is its rather slow runtime. We only see this as a small disadvantage because with bit more work the decomposition algorithms can be implemented in parallel fashion on the Graphics Processing Unit (GPU). This would lead to big improvements regarding the runtime of the decomposition algorithms. A second solution to this problem would be to exchange the sparse decomposition in the test scenario for a solver optimized for fast sparse inference. Concluding, there exist a variety of state-of-the-art SR systems and depending on the application and the type of images one can choose the appropriate method. We showed that our SR systems perform well on the tested images with different drawbacks and advantages over the other.

5. Conclusion

Contents

5.1. Summary

This thesis covered a comprehensive review of SR methods and showed some fundamental algorithms for solving l_1-regularized optimization problems often applied in such tasks. We introduced sparse coding as a state-of-the-art method for SR and illustrated the benefits and drawback of the jointly trained sparse representation scheme developed by Yang et al.[YWHM10]. We took this work as a starting point and improved their training scheme by embedding it in a bilevel formulation. We showed the derivation of a bilevel program from the model to the implementation and concluded this work with a comprehensive evaluation and comparison.

In the review of SR systems we first presented basic and advanced image interpolation algorithms. We moved to SR methods based on the inverse problem formulation followed by example based approaches. Recent example based systems like Neighborhood Embedding (NE) lead to state-of-the-art results but needed a large dataset in storage. As a method to solve this drawback, SR via sparse representation was presented. The seminal work of Yang et al. [YWHM10] which first introduced sparse coding for SR relied on a sub-optimal training scheme. This was a major motivation for this thesis. For comparison we reviewed other SR methods utilizing sparse representation like [ZEP12] or [TDG13]. Since convex optimization plays an important role in our and other's SR methods, we presented the basic solvers used in our system.

The main point of this thesis was the derivation of the bilevel training scheme from the model to an applicable algorithm. This process needed careful attention regarding the model and the used norms. This was an important lesson learned during this thesis. For example, the sparse decomposition preceding the dictionary update stage has to closely follow the underlying model to achieve convergence. Any solver which is not based on the exact model e.g. Orthogonal Matching Pursuit (OMP) or the LASSO with the regularization in the constraint, simply can not solve the problem modeled. This argument is one reason why our program is slow compared to other sparse solvers which make significant simplification for the benefit of a faster run-time. In the testing stage, for comparison we exchanged the sparse decomposition solver (FISTA) for one with a faster run-time (LASSO), but the quality of the results were not as good. In general a model should be versatile but specific enough to account for the practical situations in use. In our case, the use of a strongly convex regularization yielded a simple but computationally challenging algorithm. The results achieved by this algorithm can outperform the state-of-the-art SR systems and show the benefit of convex optimization.

The bilevel program presented in chapter 3 solves the training of two connected dictionaries. The main benefit of this bilevel optimization procedure is its optimality in the two feature spaces individually, the LR feature space and the HR feature space. With decent simplification we could apply this training scheme to l_1-regularized lower-level problem statement. The resulting training scheme benefits from the simplification in terms of the run-time with minor drawbacks in regard of the quality of the results.

We have applied our bilevel optimization program to upscale digital images. Qualitative and subjective evaluation was performed. We took a comprehensive evaluation dataset and compared our algorithms to other SR systems based on sparse coding. Due to a lack of time we could not run a full evaluation of the parameter space and their influence on the results. We instead chose parameter values like patch size or dictionary size based on available literature and comparable systems. For initialization of our algorithm we took dictionaries trained with [YWHM10] and could achieve improved results.

5.2. Further Work

Single image SR based on Sparse Coding (SC) operate on patches rather than hole images. We know from literature and from our own evaluation, that patch based systems decrease their performance at patch borders specifically when higher patch sizes are used. The patch-based system of Freeman et al.[FJP02] adds a probabilistic model to take the spatial neighborhood into account and thus they could increase the performance of estimated patches. We think that a related global strategy could also improve our training scheme. Zeyde et al.[ZEP12] showed a simple reformulation of dictionary learning problem to account for spatial neighborhood without changing the patch-based dictionary learning scheme. They added a patch-extraction operator in the problem formulation and could consequently transform the problem to a global training scheme where the dictionaries were learned on hole images. Such a formulation could also be applied to our training scheme and improve our results as this could better reflect errors at patch borders. A global training scheme also enables a system to be better trained on specific images.

A minor drawback of our bilevel program is its rather slow run-time compared to methods like [TDG13] or [ZEP12]. For the training stage, this should not be a problem, since we can compute it off-line but for testing, a fast system is preferred. We think of this only as a minor disadvantage since this problem can be easily overcome by implementing FISTA and IPIASCO in parallel fashion on the GPU. Such an implementation should lead to big run-time improvements and lead the sparse decomposition stages to state-of-the-art performance regarding the run-time.

Sparse Coding has already been applied to various tasks including image reconstruction, image denoising, image deblurring[CDMBP11], inverse half-toning[MBP12] or artistic transforms[WZLP12b]. Most of these tasks can be modeled as a bilevel program and solved with our derivations, especially image deblurring, inverse half-toning and artistic transforms. All of these methods utilize two connected dictionaries with a common sparse vector. In principle, problems within two feature spaces modeled as sparse representation can be solved by our model with minimal changes.

5.3. Conclusion

Concluding, we presented a SR method that exploits the power of bilevel programming for dictionary learning. This is especially evident for SR with higher magnification factors. Furthermore, modeling optimization problems with strictly convex functions yield state-of-the-art results with all the benefits shipped with convex optimization.

Appendix A.

Acronyms

Acronyms

BP	Belief Propagation
FISTA	Fast Iterative Shrinkage-Thresholding Algorithm
FSIM	Feature Structural Similarity
GPU	Graphics Processing Unit
GSM	Gradient Similarity
HR	High Resolution
IPIASCO	Inertial Proximal Algorithm for strongly convex Optimization
IQA	Image quality assessment
ISTA	Iterative Shrinkage-Thresholding Algorithm
LARS	Least-angle Regression
LASSO	Least Absolute Shrinkage and Selection Operator
LBFGS	Limited Broyden-Fletcher-Goldfarb-Shanno
LLE	Locally Linear Embedding

LR	Low Resolution
LS	Least-Squares
MAP	Maximum A-posteriori Pobability
ML	Maximum Likelihood
MR	Mid Resolution
MRF	Markov Random Field
NE	Neighborhood Embedding
NEDI	New Edge Directed Interpolation
NN	Nearest Neighbors
NNLS	Non-Negative Least Square
OMP	Orthogonal Matching Pursuit
PCA	Principal Component Analysis
PSNR	Peak-Signal-to-Noise Ratio
QCQP	Quadratically Constrained Quadratic Programming
RLLR	Regularized Local Linear Regression
RMS	Root Mean Square Error
RR	Ridge Regression
SAI	Soft-decision Adaptive Interpolation
SC	Sparse Coding
SPAMS	Sparse Modeling Software
SR	Super Resolution
SSIM	Structural Similarity
SVD	Singular Value Decomposition
SVM	Support Vector Machine

TV	Total Variation

Appendix B.

Tables and Figures

system:	Bicubic		Yang et al.		Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
gnd02	28.72	0.887	28.72	0.887	30.52	0.924	30.76	0.927	**31.55**	**0.932**	31.45	0.929
gnd03	28.65	0.831	28.65	0.831	29.79	0.873	29.85	0.876	**30.03**	**0.877**	29.96	0.870
gnd04	27.19	0.866	27.19	0.866	28.35	0.900	28.36	0.901	**28.59**	**0.905**	28.59	0.901
gnd05	27.35	0.889	27.35	0.889	28.97	0.922	28.96	0.922	29.43	**0.927**	**29.44**	0.924
gnd06	29.98	0.850	29.98	0.850	31.14	0.884	31.16	**0.886**	**31.30**	0.885	31.16	0.877
gnd07	30.22	0.919	30.22	0.919	32.06	0.943	32.04	0.943	**32.60**	**0.945**	32.54	0.942
gnd08	28.51	0.896	28.51	0.896	29.70	0.923	29.77	**0.925**	**29.95**	0.924	29.94	0.921
gnd09	30.53	0.922	30.53	0.922	32.44	0.948	32.29	0.945	**32.90**	**0.951**	32.83	0.949
gnd10	27.68	0.856	27.68	0.856	29.58	0.898	29.41	0.894	**30.07**	**0.904**	30.05	0.900
gnd12	29.69	0.841	29.69	0.841	31.03	0.883	31.22	0.889	**31.30**	**0.890**	31.16	0.883
gnd13	26.58	0.806	26.58	0.806	27.62	0.849	27.67	**0.853**	**27.70**	0.853	27.64	0.845
gnd14	26.61	0.765	26.61	0.765	27.51	0.816	**27.64**	**0.823**	27.64	0.823	27.60	0.812
gnd15	29.42	0.834	29.42	0.834	30.36	0.872	**30.45**	**0.876**	30.40	0.874	30.28	0.865
gnd16	31.19	0.872	31.19	0.872	32.08	0.898	**32.11**	**0.900**	32.07	0.894	31.94	0.889
gnd17	26.59	0.788	26.59	0.788	27.65	0.842	27.78	0.849	**27.88**	**0.849**	27.82	0.839
gnd18	24.76	0.772	24.76	0.772	25.61	0.820	25.71	0.826	**25.81**	**0.827**	25.78	0.819
gnd19	28.35	0.811	28.35	0.811	29.48	0.857	29.66	0.863	**29.76**	**0.864**	29.71	0.856
gnd20	29.46	0.864	29.46	0.864	30.35	0.896	30.46	**0.900**	**30.50**	0.899	30.39	0.892
gnd21	32.71	0.910	32.71	0.910	34.49	0.934	34.68	**0.936**	**34.85**	0.933	34.82	0.930
gnd23	31.45	0.938	31.45	0.938	34.18	**0.956**	34.04	0.956	34.56	0.954	**34.89**	0.955
gnd24	31.65	0.938	31.65	0.938	33.62	0.957	33.90	**0.959**	34.11	0.956	**34.15**	0.955
gnd25	42.43	0.979	42.43	0.979	43.53	0.984	**43.81**	**0.984**	42.47	0.977	42.47	0.977
gnd26	33.13	0.944	33.13	0.944	34.96	0.961	**35.07**	**0.962**	35.00	0.960	35.00	0.958
gnd27	33.07	0.916	33.07	0.916	34.46	0.939	**34.60**	**0.941**	34.57	0.934	34.49	0.930
gnd28	41.78	0.974	41.78	0.974	43.67	0.982	**44.10**	**0.983**	42.41	0.977	41.96	0.975
gnd29	38.75	0.973	38.75	0.973	40.92	0.980	**40.99**	**0.981**	40.19	0.974	39.96	0.975
gnd30	23.01	0.669	23.01	0.669	23.49	0.724	23.56	0.732	**23.58**	**0.734**	23.57	0.721
gnd31	28.08	0.807	28.08	0.807	29.15	0.845	29.06	0.844	**29.34**	**0.847**	29.30	0.841
gnd33	31.62	0.928	31.62	0.928	33.59	0.955	33.46	0.953	**34.00**	**0.959**	33.90	0.955
gnd34	26.31	0.805	26.31	0.805	27.35	0.846	27.34	0.847	**27.60**	**0.852**	27.53	0.845
gnd35	30.32	0.896	30.32	0.896	31.83	0.928	31.69	0.926	**31.90**	**0.930**	31.83	0.927
gnd36	28.23	0.863	28.23	0.863	29.58	0.900	29.62	0.902	**29.92**	**0.908**	29.83	0.901
gnd37	26.34	0.849	26.34	0.849	27.75	0.891	27.51	0.886	**27.97**	**0.899**	27.96	0.895
gnd38	26.21	0.785	26.21	0.785	27.29	0.841	27.40	0.847	**27.45**	**0.850**	27.39	0.841
gnd39	21.25	0.776	21.25	0.776	22.67	0.841	22.64	0.841	**23.00**	**0.855**	22.99	0.850
gnd40	25.82	0.816	25.82	0.816	27.42	0.865	27.41	0.866	27.79	**0.872**	**27.80**	0.866
gnd41	30.95	0.853	30.95	0.853	32.21	0.893	32.45	**0.898**	**32.50**	0.893	32.28	0.884
gnd42	32.83	0.880	32.83	0.880	33.61	0.902	**33.70**	**0.905**	33.52	0.900	33.39	0.894
gnd43	26.07	0.705	26.07	0.705	26.56	0.754	**26.62**	**0.761**	26.61	0.760	26.56	0.748
gnd45	36.73	0.969	36.73	0.969	38.94	0.978	**39.15**	**0.979**	38.84	0.974	38.84	0.973
gnd46	33.35	0.932	33.35	0.932	34.91	0.952	**35.17**	**0.955**	34.65	0.949	34.66	0.946
gnd47	35.80	0.944	35.80	0.944	37.78	0.962	**37.83**	**0.963**	37.74	0.960	37.56	0.957
gnd48	37.55	0.976	37.55	0.976	41.26	0.985	**41.34**	**0.986**	39.81	0.977	40.03	0.978
gnd49	27.77	0.808	27.77	0.808	28.90	0.852	**28.97**	**0.856**	28.95	0.853	28.91	0.844
gnd50	29.81	0.836	29.81	0.836	30.76	0.876	**30.89**	**0.881**	30.81	0.878	30.71	0.869
gnd52	25.86	0.781	25.86	0.781	26.77	0.833	**26.89**	0.839	26.86	**0.842**	26.85	0.832
gnd53	34.13	0.935	34.13	0.935	36.22	0.959	36.34	0.960	**36.73**	**0.963**	36.39	0.959
gnd54	37.43	0.950	37.43	0.950	39.53	0.967	**39.65**	**0.968**	39.60	0.964	39.33	0.961
gnd55	29.97	0.875	29.97	0.875	31.24	0.912	31.46	0.918	**31.50**	**0.919**	31.43	0.913
gnd56	28.72	0.878	28.72	0.878	30.03	0.913	30.14	0.916	**30.18**	**0.916**	30.15	0.911
gnd57	25.04	0.829	25.04	0.829	26.44	0.877	26.53	0.881	26.71	**0.887**	**26.71**	0.881
gnd58	28.72	0.878	28.72	0.878	30.33	0.914	30.36	0.917	30.47	**0.918**	**30.48**	0.913
gnd59	32.24	0.931	32.24	0.931	34.20	0.956	34.28	**0.957**	**34.39**	0.955	34.33	0.953
gnd60	27.98	0.898	27.98	0.898	29.76	0.932	29.92	0.935	30.18	**0.939**	**30.23**	0.937
gnd61	24.34	0.831	24.34	0.831	25.75	0.872	25.83	0.874	26.27	**0.884**	**26.29**	0.881
gnd63	32.34	0.960	32.34	0.960	36.43	0.977	36.21	**0.978**	36.93	0.976	**37.56**	0.976
gnd64	30.82	0.891	30.82	0.891	32.22	0.915	**32.38**	**0.917**	32.25	0.914	32.36	0.910
gnd65	26.20	0.872	26.20	0.872	28.11	0.913	27.98	0.914	28.92	**0.921**	**29.11**	0.920
gnd66	31.73	0.939	31.73	0.939	34.57	0.963	34.77	**0.964**	34.58	0.961	**34.95**	0.961
gnd67	29.16	0.891	29.16	0.891	31.50	0.927	31.53	0.928	32.03	**0.931**	**32.19**	0.927
gnd68	26.55	0.897	26.55	0.897	28.61	0.933	28.78	**0.936**	29.15	0.934	**29.34**	0.932
gnd69	26.90	0.868	26.90	0.868	28.88	0.906	28.92	0.908	29.41	**0.916**	**29.54**	0.912
gnd70	24.77	0.852	24.77	0.852	27.08	0.899	27.11	0.901	27.60	**0.903**	**27.66**	0.899
gnd71	27.86	0.876	27.86	0.876	29.52	0.913	29.61	0.915	**29.99**	**0.918**	29.92	0.913
gnd72	26.77	0.867	26.77	0.867	28.49	0.912	28.62	0.915	**29.16**	**0.925**	29.11	0.920
gnd73	28.89	0.917	28.89	0.917	30.61	0.941	30.60	0.942	30.97	**0.946**	**31.14**	0.945
gnd74	27.11	0.852	27.11	0.852	28.35	0.891	**28.45**	0.895	28.45	**0.895**	28.45	0.889
gnd75	29.91	0.905	29.91	0.905	31.87	0.938	31.97	0.941	32.59	**0.943**	**32.63**	0.938
gnd76	31.34	0.908	31.34	0.908	33.01	0.939	33.21	0.942	**33.34**	**0.945**	33.23	0.940
gnd77	24.53	0.715	24.53	0.715	25.27	0.775	**25.35**	0.782	25.34	**0.786**	25.35	0.775
gnd79	26.61	0.778	26.61	0.778	27.61	0.826	27.66	0.829	**27.81**	**0.832**	27.71	0.823
gnd80	30.33	0.884	30.33	0.884	31.90	0.920	32.03	0.923	**32.45**	**0.927**	32.30	0.922
average LASSO	29.59	0.869	29.59	0.869	31.16	0.904	31.23	0.906	**31.35**	**0.907**	31.33	0.902
mean run-time [s],LASSO	-	-	0	-	4.8	-	**0.4**	-	5.2	-	5.3	-

Table B.1.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] on the test set “Li He” from [TDG13] for a scaling factor of 2. We used the solver of “cite spams” in combination with our trained dictionaries.

system:	Bicubic		Yang et al.		Zeyde et al.		Timofte et al.		Our Active Set		Our Smoothed l1e	
image	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
gnd02	26.23	0.796	26.23	0.796	27.18	0.835	27.19	0.834	27.50	0.842	**27.62**	**0.843**
gnd03	26.50	0.723	26.50	0.723	27.25	0.764	27.23	0.766	27.35	**0.766**	**27.38**	0.759
gnd04	24.99	0.763	24.99	0.763	25.67	0.799	25.66	0.797	25.69	**0.801**	**25.71**	0.797
gnd05	24.51	0.787	24.51	0.787	25.48	0.825	25.41	0.823	25.59	**0.830**	**25.67**	0.826
gnd06	27.46	0.739	27.46	0.739	28.21	0.776	28.17	**0.777**	28.22	0.774	**28.22**	0.765
gnd07	27.32	0.842	27.32	0.842	28.45	0.870	28.37	0.868	28.63	**0.872**	**28.73**	0.869
gnd08	25.93	0.809	25.93	0.809	26.79	0.844	26.78	0.844	27.02	**0.849**	**27.20**	0.846
gnd09	27.41	0.839	27.41	0.839	28.53	0.875	28.39	0.869	**28.71**	**0.880**	28.67	0.878
gnd10	24.65	0.737	24.65	0.737	25.74	0.783	25.52	0.773	25.94	0.787	**26.15**	**0.791**
gnd12	27.01	0.706	27.01	0.706	27.81	0.752	27.89	**0.759**	27.94	0.757	**27.98**	0.748
gnd13	24.03	0.653	24.03	0.653	24.62	0.700	24.65	**0.704**	24.57	0.701	**24.68**	0.691
gnd14	24.37	0.603	24.37	0.603	24.93	0.655	**24.99**	**0.662**	24.94	0.656	24.96	0.642
gnd15	26.97	0.701	26.97	0.701	27.52	0.742	**27.57**	**0.748**	27.49	0.740	27.44	0.727
gnd16	29.09	0.789	29.09	0.789	29.67	0.815	**29.68**	**0.818**	29.59	0.808	29.59	0.800
gnd17	24.31	0.625	24.31	0.625	24.81	0.674	24.87	**0.682**	**24.91**	0.677	24.89	0.664
gnd18	22.78	0.632	22.78	0.632	23.17	0.670	23.18	**0.673**	**23.24**	0.673	23.22	0.664
gnd19	26.01	0.669	26.01	0.669	26.63	0.715	26.69	**0.721**	**26.73**	0.719	26.70	0.705
gnd20	27.38	0.768	27.38	0.768	27.85	0.799	**27.89**	**0.802**	27.88	0.799	27.85	0.791
gnd21	29.95	0.838	29.95	0.838	31.20	0.866	31.23	**0.868**	31.41	0.866	**31.50**	0.863
gnd23	28.28	0.883	28.28	0.883	30.24	0.912	30.09	0.910	30.66	**0.912**	**31.18**	0.912
gnd24	28.69	0.879	28.69	0.879	29.86	0.902	29.94	**0.903**	30.05	0.902	**30.20**	0.900
gnd25	39.03	0.957	39.03	0.957	41.20	0.972	**41.37**	**0.973**	39.94	0.961	39.41	0.955
gnd26	29.76	0.882	29.76	0.882	31.28	0.912	31.28	**0.912**	31.29	0.911	**31.43**	0.908
gnd27	30.38	0.842	30.38	0.842	31.19	0.866	**31.27**	**0.869**	31.22	0.858	31.17	0.850
gnd28	37.70	0.941	37.70	0.941	39.14	0.954	**39.40**	**0.956**	38.20	0.946	37.99	0.941
gnd29	35.67	0.951	35.67	0.951	**37.48**	0.961	37.45	**0.961**	36.73	0.953	36.84	0.951
gnd30	21.69	0.520	21.69	0.520	21.95	0.564	**21.99**	**0.572**	21.98	0.567	21.97	0.553
gnd31	25.95	0.692	25.95	0.692	26.66	0.727	26.55	0.724	26.76	**0.727**	**26.81**	0.723
gnd33	28.35	0.836	28.35	0.836	29.79	0.880	29.73	0.879	29.89	**0.884**	**29.95**	0.880
gnd34	24.35	0.694	24.35	0.694	24.84	0.726	24.82	0.726	24.89	**0.726**	**24.99**	0.726
gnd35	27.63	0.796	27.63	0.796	28.77	**0.840**	28.60	0.836	28.67	0.839	**28.78**	0.837
gnd36	25.61	0.746	25.61	0.746	26.44	0.789	26.43	0.790	26.54	**0.796**	**26.67**	0.792
gnd37	23.76	0.719	23.76	0.719	24.56	0.764	24.46	0.759	24.59	**0.771**	**24.65**	0.765
gnd38	23.60	0.595	23.60	0.595	24.12	0.653	24.16	**0.661**	**24.17**	0.660	24.15	0.645
gnd39	18.38	0.561	18.38	0.561	19.00	0.623	18.97	0.622	19.07	**0.640**	**19.13**	0.634
gnd40	23.10	0.669	23.10	0.669	24.13	0.724	24.07	0.723	24.32	**0.733**	**24.48**	0.732
gnd41	28.37	0.725	28.37	0.725	28.91	0.757	**28.97**	**0.761**	28.95	0.751	28.86	0.738
gnd42	30.73	0.804	30.73	0.804	31.24	0.826	**31.31**	**0.829**	31.06	0.818	31.01	0.811
gnd43	24.66	0.567	24.66	0.567	24.95	0.610	**24.97**	**0.617**	24.94	0.608	24.90	0.594
gnd45	32.71	0.932	32.71	0.932	34.08	0.945	**34.28**	**0.946**	34.06	0.940	34.14	0.936
gnd46	29.45	0.837	29.45	0.837	30.58	0.870	**30.67**	**0.873**	30.43	0.868	30.43	0.860
gnd47	32.28	0.877	32.28	0.877	33.60	0.903	33.56	**0.904**	33.59	0.900	**33.60**	0.896
gnd48	32.81	0.935	32.81	0.935	**35.70**	**0.954**	35.53	0.954	35.18	0.943	35.65	0.943
gnd49	25.32	0.675	25.32	0.675	26.01	0.716	26.02	**0.720**	26.01	0.713	**26.04**	0.704
gnd50	27.40	0.695	27.40	0.695	27.97	0.742	**28.04**	**0.750**	27.94	0.739	27.86	0.724
gnd52	23.42	0.600	23.42	0.600	23.91	0.658	**23.99**	**0.667**	23.93	0.667	23.92	0.650
gnd53	30.95	0.864	30.95	0.864	32.11	0.895	32.21	0.898	**32.31**	**0.901**	32.22	0.895
gnd54	34.14	0.895	34.14	0.895	35.52	0.918	**35.57**	**0.920**	35.54	0.914	35.33	0.908
gnd55	27.12	0.742	27.12	0.742	27.91	0.792	**28.03**	**0.800**	27.97	0.798	27.94	0.785
gnd56	25.80	0.752	25.80	0.752	26.51	0.795	**26.62**	**0.800**	26.58	0.798	26.61	0.790
gnd57	22.39	0.677	22.39	0.677	23.22	0.729	23.25	0.733	23.31	**0.739**	**23.40**	0.733
gnd58	25.84	0.759	25.84	0.759	27.08	0.808	27.09	0.813	27.11	**0.814**	**27.35**	0.809
gnd59	28.80	0.841	28.80	0.841	29.99	0.878	30.00	0.880	30.09	**0.880**	**30.10**	0.874
gnd60	24.62	0.772	24.62	0.772	25.61	0.818	25.71	0.823	25.79	**0.828**	**25.88**	0.824
gnd61	21.76	0.713	21.76	0.713	22.71	0.761	22.73	0.762	22.96	0.778	**23.08**	**0.779**
gnd63	28.01	0.896	28.01	0.896	30.47	0.928	30.44	0.928	31.06	0.930	**31.73**	**0.931**
gnd64	27.84	0.807	27.84	0.807	29.17	0.840	29.28	**0.844**	29.21	0.841	**29.46**	0.836
gnd65	23.34	0.739	23.34	0.739	24.39	0.789	24.30	0.788	24.80	**0.803**	**24.81**	0.801
gnd66	27.50	0.853	27.50	0.853	29.31	0.890	29.47	0.892	29.46	0.892	**29.85**	**0.893**
gnd67	25.53	0.776	25.53	0.776	26.98	0.822	27.17	0.826	27.37	**0.830**	**27.55**	0.826
gnd68	23.28	0.771	23.28	0.771	24.40	0.813	24.61	**0.819**	24.80	0.819	**25.07**	0.815
gnd69	23.78	0.758	23.78	0.758	25.18	0.810	25.27	0.813	25.56	**0.825**	**25.75**	0.824
gnd70	21.53	0.703	21.53	0.703	22.91	0.761	22.88	0.762	23.22	**0.769**	**23.39**	0.769
gnd71	24.97	0.759	24.97	0.759	25.88	0.799	25.87	**0.801**	26.02	0.801	**26.11**	0.795
gnd72	23.80	0.726	23.80	0.726	24.75	0.781	24.83	0.785	24.96	**0.795**	**25.02**	0.789
gnd73	25.93	0.840	25.93	0.840	26.77	0.866	26.74	0.867	26.87	**0.872**	**26.88**	0.869
gnd74	24.29	0.716	24.29	0.716	24.96	0.759	**25.03**	**0.764**	24.97	0.762	25.02	0.754
gnd75	26.59	0.790	26.59	0.790	27.69	0.832	27.70	0.834	27.83	**0.836**	**27.92**	0.830
gnd76	28.14	0.801	28.14	0.801	29.10	0.840	29.21	0.845	29.20	**0.846**	**29.22**	0.841
gnd77	22.62	0.538	22.62	0.538	23.04	0.595	23.08	0.604	**23.09**	**0.605**	23.09	0.589
gnd79	24.31	0.625	24.31	0.625	24.91	0.671	24.91	**0.674**	24.97	0.673	**25.01**	0.664
gnd80	27.68	0.781	27.68	0.781	28.67	0.824	28.70	0.826	28.88	**0.830**	**28.91**	0.824
average LASSO	26.76	0.760	26.76	0.760	27.75	0.799	27.78	0.801	27.81	**0.802**	**27.88**	0.795
mean run-time [s],LASSO	-	-	0.0	-	2.2	-	**0.3**	-	3.3	-	3.3	-

Table B.2.: This table shows the evaluation of our bilevel sparse coding algorithms compared to the works of Yang et al.[YWHM10], Zeyde et al.[ZEP12] and Timofte et al.[TDG13] on the test set "Li He" from [TDG13] for a scaling factor of 2. We used the solver of "cite spams" in combination with our trained dictionaries.

Bibliography

[AD05] H. A. Aly and E. Dubois. Image up-sampling using total-variation regularization with a new observation model. *Trans. Img. Proc.*, 14(10):1647–1659, October 2005.

[AEB06] M. Aharon, M. Elad, and A Bruckstein. K -svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, Nov 2006.

[Bar98] J.F. Bard. *Practical Bilevel Optimization: Algorithms and Applications*. Nonconvex Optimization and Its Applications. Springer, 1998.

[BJMO12] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

[BK02] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1167–1183, Sep 2002.

[BLNZ95] R. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[BM73] Jerome Bracken and James T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.

[BM87] G. J. Burton and Ian R. Moorhead. Color and spatial structure in natural scenes. *Appl. Opt.*, 26(1):157–170, Jan 1987.

[BRGA12] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *British Machine Vision Conference, BMVC 2012, Surrey, UK, September 3-7, 2012*, pages 1–10, 2012.

[BT09] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202, March 2009.

[CDMBP11] Florent Couzinie-Devy, Julien Mairal, Francis Bach, and Jean Ponce. Dictionary learning for deblurring and digital zoom. *Computing Research Repository*, abs/1110.0957, 2011.

[CMS07] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, September 2007.

[Com99] Eastman Kodak Company. Photocd PCD0992. `http://r0k.us/graphics/kodak/`, 1999. Last visited on 10/10/2014.

[CP11] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[CW08] E.J. Candes and M.B. Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, March 2008.

[CYX04] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. Super-resolution through neighbor embedding. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–I, June 2004.

[Dem02] S. Dempe. *Foundations of Bilevel Programming*. Nonconvex Optimization and Its Applications. Springer, 2002.

[DLHT14] Chao Dong, ChenChange Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014*, volume 8692 of *Lecture Notes in Computer Science*, pages 184–199. Springer International Publishing, 2014.

[DZLS13] Weisheng Dong, Lei Zhang, Rastislav Lukac, and Guangming Shi. Sparse representation based image interpolation with nonlocal autoregressive modeling. *IEEE Transactions on Image Processing*, 22(4):1382–1394, 2013.

[EA06] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12):3736–3745, Dec 2006.

[EF97] M. Elad and A. Feuer. Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images. *IEEE Transactions on Image Processing*, 6(12):1646–1658, Dec 1997.

[EHJT04] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 04 2004.

[Fat07] Raanan Fattal. Image upsampling via imposed edge statistics. *ACM Trans. Graph.*, 26(3), July 2007.

[FJP02] W.T. Freeman, T.R. Jones, and E.C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, Mar 2002.

[Fol96] J.D. Foley. *Computer Graphics: Principles and Practice*. Addison-Wesley systems programming series. Addison-Wesley, 1996.

[FR11] Massimo Fornasier and Holger Rauhut. Compressive sensing. In Otmar Scherzer, editor, *Handbook of Mathematical Methods in Imaging*, pages 187–228. Springer New York, 2011.

[FREM04] S. Farsiu, M.D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super resolution. *Image Processing, IEEE Transactions on*, 13(10):1327–1344, 2004.

[HQZ13] Li He, Hairong Qi, and R. Zaretzki. Beta process joint dictionary learning for coupled feature spaces with application to single image super-resolution. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 345–352, June 2013.

[HT84] T. S. Huang and R. Y. Tsai. Multi-frame image restoration and registration. *Adv Computer Vision Image Process*, 1:317–339, 1984.

[IP91] Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graphical Models and Image Processing*, 53(3):231 – 239, 1991.

[IP13] Peter Innerhofer and Thomas Pock. A convex approach for image hallucination. *Proceedings 37th Workshop of the Austrian Association for Pattern Recognition*, abs/1304.7153, 2013.

[KH12] K. Keong Chua and Y. Haur Tay. On the Adaptability of Neural Network Image Super-Resolution. *ArXiv e-prints*, December 2012.

[LBRN06] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[LLN12] Anmin Liu, Weisi Lin, and M. Narwaria. Image quality assessment based on gradient similarity. *Image Processing, IEEE Transactions on*, 21(4):1500–1512, April 2012.

[LN89] DongC. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

[LO00] Xin Li and M.T. Orchard. New edge directed interpolation. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, volume 2, pages 311–314 vol.2, Sept 2000.

[LZX+11] Xianming Liu, Debin Zhao, Ruiqin Xiong, Siwei Ma, Wen Gao, and Huifang Sun. Image interpolation via regularized local linear regression. *IEEE Transactions on Image Processing*, 20(12):3455–3469, Dec 2011.

[MBP12] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):791–804, April 2012.

[MBPS09] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 689–696, New York, NY, USA, 2009. ACM.

[MN99] Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. Wiley, 1999.

[MPSC09] Dennis Mitzel, Thomas Pock, Thomas Schoenemann, and Daniel Cremers. Video super resolution using duality based tv-l1 optical flow. In Joachim Denzler, Gunther Notni, and Herbert Süße, editors, *Pattern Recognition*, volume 5748 of *Lecture Notes in Computer Science*, pages 432–441. Springer Berlin Heidelberg, 2009.

[MSS04] Patrice Marcotte, Gilles Savard, and Frédéric Semet. A bilevel programming approach to the travelling salesman problem. *Operations Research Letters*, 32(3):240 – 248, 2004.

[OBP14] P. Ochs, T. Brox, and T. Pock. ipiasco: Inertial proximal algorithm for strongly convex optimization. *Technical Report*, 2014.

[PB14] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3), 2014.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Representation and Reasoning Series. Morgan Kaufmann, 1988.

[Pol87] Boris T Poljak. *Introduction to optimization*. Optimization Software, 1987.

[RBE10] R. Rubinstein, A.M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, June 2010.

[RZE08] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. Efficient implementation of the k-SVD algorithm using batch orthogonal matching pursuit. *CS Technion*, page 40, 2008.

[SH12] Wan-Chi Siu and Kwok-Wai Hung. Review of image interpolation and super-resolution. In *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–10, Dec 2012.

[SSXS08] Jian Sun, Jian Sun, Zongben Xu, and Heung-Yeung Shum. Image super-resolution using gradient profile prior. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[TA77] A.N. Tikhonov and V.I.A. Arsenin. *Solutions of ill-posed problems*. Scripta series in mathematics. Winston, 1977.

[TDG13] Radu Timofte, Vincent De, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1920–1927, Dec 2013.

[TDSVG14] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. *Asian Conference on Computer Vision (ACCV 2014)*, 2014.

[UPWB10] M. Unger, T. Pock, M. Werlberger, and H. Bischof. A convex approach for variational super-resolution. *Proceedings 32th DAGM Pattern Recognition Symposium"*, pages 313–322, 2010.

[VC94] LuísN. Vicente and PaulH. Calamai. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5(3):291–306, 1994.

[WBSS04] Zhou Wang, AC. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, April 2004.

[Wei14] Eric W. Weisstein. Newton's method. From MathWorld—A Wolfram Web Resource. `http://mathworld.wolfram.com/NewtonsMethod.html`, 2014. Last visited on 10/10/2014.

[WZLP12a] Shenlong Wang, D. Zhang, Yan Liang, and Quan Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2216–2223, June 2012.

[WZLP12b] Shenlong Wang, Lei Zhang, Yan Liang, and Quan Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2216–2223. IEEE, 2012.

[YFW00] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 13, pages 689–695, 2000.

[YGZ+10] Allen Y. Yang, Arvind Ganesh, Zihan Zhou, Shankar Sastry, and Yi Ma. A review of fast l1-minimization algorithms for robust face recognition. *CoRR*, abs/1007.3753, 2010.

[YH11] Jianchao Yang and Thomas Huang. *Image super-resolution: historical overview and future challenges.* from the book: Super-Resolution Imaging (edited by Peyman Milanfar). CRC Press (Taylor & and amp and Francis Group), 2011.

[YWHM10] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, November 2010.

[YWL+12a] Jianchao Yang, Zhaowen Wang, Zhe Lin, Scott Cohen, and Thomas Huang. Coupled dictionary training for image super-resolution. *Image Processing, IEEE Transactions on*, 21(8):3467–3478, 2012.

[YWL+12b] Jianchao Yang, Zhaowen Wang, Zhe Lin, Xianbiao Shu, and Thomas Huang. Bilevel sparse coding for coupled feature spaces. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2360–2367. IEEE, 2012.

[ZEP12] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces*, pages 711–730. Springer, 2012.

[ZFW13] Xuexia Zhong, Guorui Feng, and Jian Wang. A double constrained soft-decision image interpolation algorithm. In *TENCON 2013 - 2013 IEEE Region 10 Conference (31194)*, pages 1–4, Oct 2013.

[ZW08] Xiangjun Zhang and Xiaolin Wu. Image interpolation by adaptive 2-d autoregressive modeling and soft-decision estimation. *IEEE Transactions on Image Processing*, 17(6):887–896, June 2008.

[ZZMZ11] Lin Zhang, D. Zhang, Xuanqin Mou, and D. Zhang. FSIM: a feature similarity index for image quality assessment. *Image Processing, IEEE Transactions on*, 20(8):2378–2386, 2011.