Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica

Fast super-resolution and segmentation based on ℓ_0 gradient minimization

Relatore:
Chiar.ma Prof.
Elena Loli Piccolomini

Presentata da:
Dario Mylonopoulos

Correlatore:
Dott.
Pasquale Cascarano

I Sessione
Anno Accademico 2020/2021

Introduction

The use of digital images is becoming more and more widespread across many different scientific and technical fields as well as within aspects of every day life. Image processing techniques are thus becoming increasingly important to improve the quality of captured images and extract the required information. Super-resolution and segmentation are two difficult problems in image processing and in the past years there has been continuous work to improve existing algorithms and find new approaches.

Our main contribution is the implementation of an efficient solution for the super-resolution problem that significantly improves the speed of a previous algorithm. We apply this new method to two joint super-resolution and segmentation models based on on ℓ_0 gradient minimization and compare the results of the algorithms.
In chapter 1 we briefly introduce digital images and the formation model of digital photographs. We then go over the super-resolution and segmentation problems and some of the fields of application.
In chapter 2 we define the numerical model of image reconstruction and its application to super-resolution.
In chapter 3 we introduce the optimization algorithm ADMM and how it is applicable to the super-resolution problem. We also outline the implementation of the new and more efficient super-resolution algorithm, and two regularization models based on ℓ_0 gradient minimization.
In chapter 4 we analyse the results of the algorithms and discuss time complexity and measured execution time.

Contents

Chapter 1

Super-resolution and segmentation

1.1 Digital images

Digital images are most commonly captured with a digital camera, but this is only one of the many possible ways. Images can be constructed for countless purposes and from many different types of signals, for example tomography techniques produce images by sections through the use of different penetrating waves, such as X-rays and ultrasound. Astronomical photography allows us to see extremely far away in the universe and electron microscopes make it possible to capture images of surfaces smaller than the wavelength of visible light. We will mainly use digital photographs in our examples as it's the type of image everyone is most familiar with, but in general we can think of a digital image as the capture of any two-dimensional signal by an image sensor that is processed by a computer and then displayed as visible light.

A digital image is stored in the memory of a computer as a two-dimensional array of numbers and it can be written as a matrix whose elements represent the intensity of a small square section called pixel. The number of pixels in an image is its resolution, higher resolutions allow for greater detail. The simplest way to store a grayscale image is with an 8 bit integer number per pixel with values between 0 and 255. Whenever we want to operate on images it's useful to represent pixels as floating point numbers with values between

0 (minimum intensity) and 1 (maximum intensity) for higher precision. To represent color images we also need more than one value for each pixel, a format commonly used is RGB: for each pixel we store 3 numbers that are referred to as channels and represent the intensity of the color red, green and blue respectively. Other colors are obtained through the sum of these 3, as it happens in LCD displays where each pixel is made of three small LEDs positioned very close to each other. Color images can thus be represented as three matrices, one for each channel.

1.2 Image formation model

When we take a picture with a digital camera light enters through the lens and is measured by a sensor made of a matrix of elements sensitive to light that correspond to the pixels of the image. The two most common types of image sensors are the Charge coupled device (CCD) and the active-pixel (CMOS) which convert the measured light to an analogical signal that is then read and stored as a digital number. The greater the number of elements of the sensor the higher will be the resolution of the captured image.

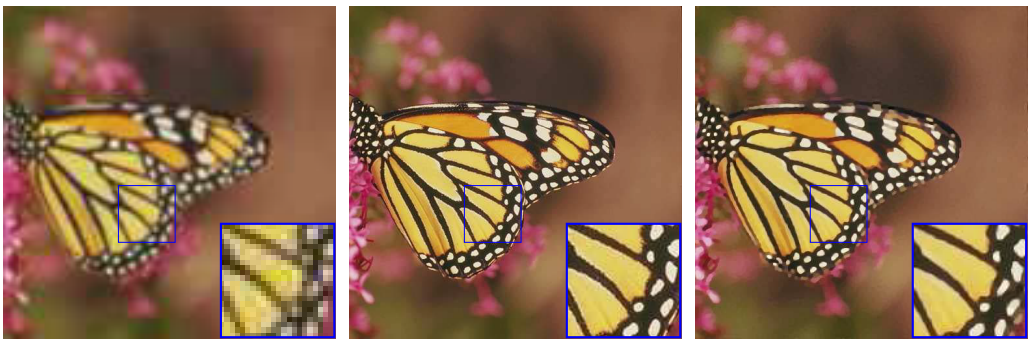
However, captured images are often subject to degradation from various sources that lower their quality. For example a photograph may appear blurred because it was captured with an out-of-focus lens, due to atmospheric conditions or because of movement of the camera. Furthermore acquired images are almost always contaminated with noise that can be of various types and come from various sources. The most common type of noise that we will consider is additive noise, that is values summed to the pixels of the original image. A typical source of noise for images captured with a camera sensor is caused by errors in the measurement of voltage values. This readout noise is usually assumed to consist of independent and identically distributed random values (white noise) with a Gaussian distribution with mean zero and standard deviation proportional to the amplitude of the noise, for this reason it is known as Additive White Gaussian Noise (AWGN).

1.3 Super-resolution

Image super-resolution refers to the process of reconstructing an image with a higher spatial resolution using low resolution observations while preserving its quality and interesting features. This is often desirable for many purposes, improving the resolution and detail of an image can be important in many fields, such as medical, astronomical, and satellite imaging. The resolution of images is usually limited by the resolution of the sensor used to capture them, larger or more dense sensors allow for higher resolutions but are obviously more expensive and not always available. Furthermore to achieve higher density, the elements of the sensor have to be smaller and thus the amount of light incident on each one decreases resulting in additional noise. For all these reasons chips and optical components to capture very high-resolution images are prohibitively expensive and not practical in most real applications. Often it's easier to address this problem by accepting the image degradations and by using signal processing techniques to post-process the captured images and use computation to save on the hardware cost. These techniques are specifically referred as super-resolution reconstruction. Some of the fields where super-resolution addresses these problems are:

Video information enhancement In this field the use of super-resolution techniques is aimed at improving the quality of video images, for example used to convert from Standard Definition TV (SDTV) to High Definition TV (HDTV). This techniques can also be employed to improve the quality of everyday pictures and videos captured with phones, tablets and computers.

Surveillance The use of video recorders (DVR) devices is becoming more and more widespread in applications such as traffic surveillance and security monitoring. It is, however, impossible to to equip large-scale high resolution devices for these purposes because of the prohibitive costs. This field is also particularly challenging because of the impact of weather conditions and other sources of degradation such as video compression.



(a) observation (b) ground truth (c) super-resolution

Figure 1.1: Example of image super-resolution

Medical diagnosis Images are used in the medical field to provide both anatomical information about the human body structure and functional information. Many of the instruments used to capture medical images have limited resolutions and are subject to different kinds of degradations. Super-resolution technologies have been used with multiple medical imagin modalities, including magnetic resonance imaging (MRI), functional MRI (fMRI), and positron emission tomography (PET).

Astronomical observation The physical resolution of astronomical imaging devices limited by system parameters also provides a chance for SR techniques to play a role. Astronomical systems can typically collect a series of images for SR. By improving the resolution of astronomical images, Super-resolution can help astronomers with the exploration of outer space.

Biometric information identification Super-resolution is also important in biometric recognition, including resolution enhancement for faces, fingerprints and images. The resolution of biometric images is pivotal in the recognition and detection process.

1.4 Segmentation

Image segmentation involves partitioning digital images into multiple segments or objects to simplify or change the representation into something that is more meaningful and easier to analyse. It can be formulated as a classification problem of pixels with semantic labels, the input is a digital image and the output is a new image of the same size in which all pixels of the same segment or class have the same value. We can divide segmentation problems in two categories: semantic segmentation and instance segmentation. Semantic segmentation is concerned with labeling pixels with a set of object categories, such as human, car, tree. Instance segmentation extends semantic segmentation by also detecting each distinct object of interest, for example by partitioning individual persons.

Image segmentation has many applications in the field of remote sensing, including techniques for land-cover classifications, urban planning and precision agriculture. Other important application fields include medical imaging, biology and evaluation of construction materials.

Numerous image segmentation algorithms have been developed, from the earliest methods such as thresholding, region growing, k-means clustering and watershed to more advanced algorithms such as active contours, graph cuts, conditional and Markov random fields and sparsity based methods.

In recent years with the rapid growth of Artificial Intelligence techniques, Deep Learning based methods have provided a new generation of segmentation models with remarkable improvements. We refer to [8] for a detailed review of Deep Learning models used for image segmentation.

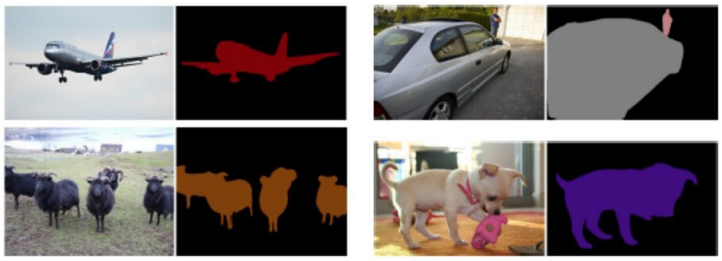
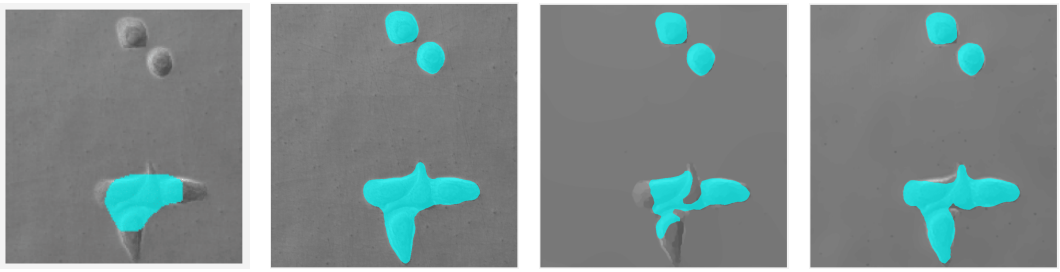


Figure 1.2: Example of semantic segmentation (results of DeepLabV3)

1.5 Joint super-resolution and segmentation

When working with low resolution images it may be useful to increase the quality of the data by applying a super-resolution algorithm before segmentation. As shown in [4], in several imaging applications, such as computed tomography, magnetic resonance and microscopy, carrying out reconstruction and partitioning tasks, jointly, has provided better results than performing the two steps sequentially. In figure 1.3 we show an example of the improvements of performing joint super-resolution and segmentation.



(a) LR (x5) + mask (b) GT + mask (c) SR + IP + mask (d) Joint + mask

Figure 1.3: (a) Segmentation of a low resolution observation. (b) Segmentation of the ground truth. (c) Segmentation applied after super-resolution. (d) Joint super-resolution and segmentation.

Chapter 2

Numerical Model

2.1 Numerical model for super-resolution

We are interested in reconstructing the original image from an observation, we can use a discrete linear model to approximate the degradation and take advantage of numerical methods to deal with this problem. We will first analyse the numerical model used for deblurring images, and we will then expand it to the problem of super-resolution.

Let $b \in \mathbb{R}^N$ be the observed image of size $N = m \times n$, where m is the number of pixels in each column and n the number of pixels in each row, and $x \in \mathbb{R}^N$ the original image of the same size. $A \in \mathbb{R}^{N \times N}$ is a matrix that describes the linear application of the blurring process and $\eta \in \mathbb{R}^N$ represents the Additive White Gaussian Noise. We can describe the effect of blurring x through A and adding the noise η to obtain b with the following formula

$$b = Ax + \eta$$

We are interested in solving the inverse problem, that is computing the original image x given the observation b and the approximation of the blurring matrix A. This is an ill-posed problem as the linear system is usually overdetermined and does not admit a solution. Therefore we look for the solution to the following least squares problem:

$$\arg\min_x \frac{1}{2}||Ax - b||_2^2$$

Although this formulation admits a solution, the results are heavily corrupted by the presence of noise in the observation. A general approach to improve these results is to add a regularization term to encode some prior knowledge about the solution x and improve stability, we thus look for a minimizer of the following function

$$\arg\min_x \frac{1}{2}||Ax - b||_2^2 + \mu R(x)$$

where $\mu > 0$ is the regularization parameter that balances the weight of the fidelity term $||Ax-b||_2^2$ and the regularization term $R(x)$. The regularization term usually aims at reducing the noise in the resulting image by increasing the smoothness of the solution, some common choices are functions of the norm of the image, the image gradient or the total variation of the image. For this reason the optimal value of the regularization parameter is dependant on the amount of noise present in the image, stronger noise requires an higher parameter to achieve the desired smoothness in the solution.

Reconstruction based super-resolution approaches expand on the image reconstruction model we have just seen by introducing the downsampling operator. As we did before we are going to solve an inverse problem, the downsampling operator discards pixels of the input image lowering its resolution. Let $\mathbf{g} \in \mathbb{R}^{N_l}$ be the low resolution observation of size $N_l = m_l \times n_l$ and $\mathbf{u} \in \mathbb{R}^{N_h}$ the high resolution original image of size $N_h = m_h \times n_h$ that we want to reconstruct. We assume the degradation model given by

$$\mathbf{g} = \mathbf{SHu} + \eta$$

where $\mathbf{H} \in \mathbb{R}^{N_h \times N_h}$ is the blurring operator, $\mathbf{S} \in \mathbb{R}^{N_l \times N_h}$ is the downsampling operator and $\eta \in \mathbb{R}^{N_l}$ the Additive White Gaussian Noise. Again we will use a minimization approach to find a solution to the inverse problem, looking for a minimizer of the following function

$$\arg\min_{\mathbf{u}} \frac{1}{2}||\mathbf{SHu} - \mathbf{g}||_2^2 + \mu R(\mathbf{u})$$

where $\mu > 0$ is the regularization parameter that balances the weight of the fidelity term $||\mathbf{SHu} - \mathbf{g}||_2^2$ and the regularization term $R(\mathbf{u})$.

We will also consider a constrained model for super-resolution given by

$$\underset{\mathbf{u}\in\mathbb{R}^{N_h}}{\arg\min}\ \frac{1}{2}||\mathbf{SHu}-\mathbf{g}||_2^2 \quad \text{s.t.} \quad R(\mathbf{u})\leq\alpha.$$

in this formulation the parameter α acts as a constraint on the value of a certain property of the result described by $R(\mathbf{u})$.

2.2 Blurring operator

The blurring operator $\mathbf{H}$ represents the linear application describing the blurring process. We are going to consider blurs that are spatially invariant and can thus be applied through convolution of the image with a blur kernel. The most common type of kernels that we are going to use is the Gaussian kernel, but the same techniques can be applied to other kernels that describe effects such as motion blur, out-of-focus lenses and atmospheric turbulence. A two-dimensional Gaussian kernel is described by the following formula

$$p_{ij}=\exp\left(-\frac{1}{2}\left(\frac{(i-k)}{\sigma}\right)^2-\frac{1}{2}\left(\frac{(j-l)}{\sigma}\right)^2\right)$$

p_{ij} represents the element at the row i and column j of the kernel, r is the blur radius, k and l are the pixel coordinates of the center of the Gaussian function. σ represents the standard deviation of the Gaussian function and affects the blur magnitude. The elements of the Gaussian kernel must be normalized so that they sum to 1 to preserve the average brightness of the input image.

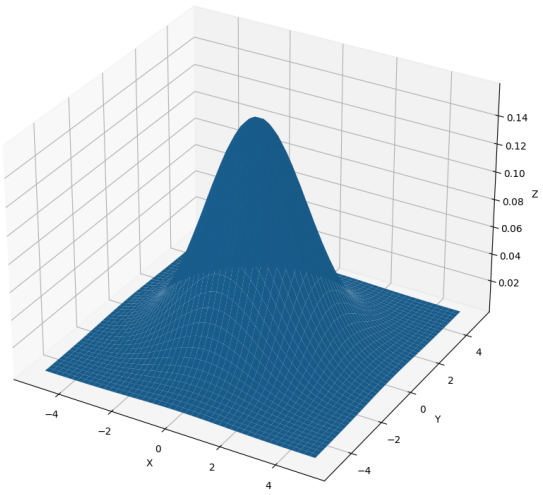


Figure 2.1: 3D plot of a Gaussian kernel with standard deviation $\sigma = 2$

If we consider periodic boundary conditions the matrix $\mathbf{H}$ is Block Circulant with Circulant Blocks (BCCB), therefore it can be decomposed through the Fourier Transform in the following way

$$\mathbf{H} = \mathbf{F}^H \mathbf{\Lambda} \mathbf{F}$$

$\mathbf{F}$ is the two-dimensional discrete Fourier transform (DFT) matrix and $\mathbf{F}^H$ is the inverse transform. $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\mathbf{H}$. This decomposition makes the computation of the blurring particularly efficient because we can use the Fast Fourier Transform (FFT) algorithm to compute the DFT and its inverse with complexity $O(N \log N)$ instead of $O(N^2)$ required for matrix-vector multiplication.

2.3 Decimation operator

The decimation operator $\mathbf{S} \in \mathbb{R}^{N_l \times N_h}$ performs downsampling of an high resolution image of size $N_h = n_h \times m_h$ and returns a low resolution image of size $N_l = n_l \times m_l$. The effect of applying the decimation operator is equivalent to discarding rows and columns of the input image. The downsampling factor

$d = d_r \times d_c$ is defined such that $N_h = N_l \times d$. The decimation factors d_r and d_c represent the discarded rows and columns of the input images respectively, satisfying the relation $m_h = m_l \times d_r$ and $n_h = n_l \times d_c$. We will also consider the conjugate transpose operator $\mathbf{S}^H \in \mathbb{R}^{N_h \times N_l}$ as the interpolation of the input image with zeros to obtain an image with higher resolution. With these definitions we have the following equality $\mathbf{S}\mathbf{S}^H = \mathbf{I}_{N_l}$.

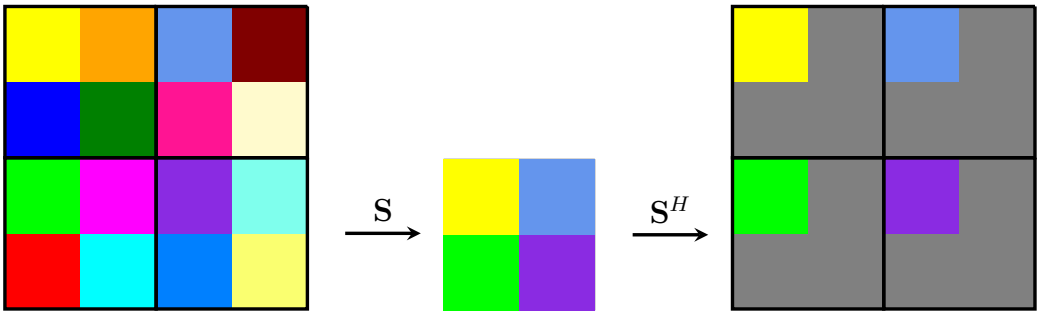


Figure 2.2: Example application of the decimation operator $\mathbf{S}$ and its conjugate $\mathbf{S}^H$.

Unfortunately, the operator $\mathbf{S}$ does not share the same properties of $\mathbf{H}$, since it cannot be diagonalized by the 2D discrete Fourier Transform. Under the previous assumptions, by denoting with $\mathbf{J}_d \in \mathbb{R}^{d \times d}$ a matrix of ones, $\mathbf{1}_d$ a d-dimensional vector of ones and $\mathbf{I}_d \in \mathbb{R}^{d \times d}$ the identity matrix, the following chain of equalities holds:

$$
\begin{aligned}
\mathbf{F}\mathbf{S}^H\mathbf{S}\mathbf{F}^H =& \frac{1}{d}\left(\mathbf{J}_{d_r} \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{J}_{d_c} \otimes \mathbf{I}_{n_c}\right) \\
=& \frac{1}{d}\left(\mathbf{1}_{d_r}\mathbf{1}_{d_r}^T \otimes \mathbf{I}_{n_r}\mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_{d_c}\mathbf{1}_{d_c}^T \otimes \mathbf{I}_{n_c}\mathbf{I}_{n_c}\right) \\
=& \frac{1}{d}\left(\left(\mathbf{1}_{d_r} \otimes \mathbf{I}_{n_r}\right)\left(\mathbf{1}_{d_r}^T \otimes \mathbf{I}_{n_r}\right)\right) \otimes \left(\left(\mathbf{1}_{d_c} \otimes \mathbf{I}_{n_c}\right)\left(\mathbf{1}_{d_c}^T \otimes \mathbf{I}_{n_c}\right)\right) \\
=& \frac{1}{d}\left(\left(\mathbf{1}_{d_r} \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_{d_c} \otimes \mathbf{I}_{n_c}\right)\right)\left(\mathbf{1}_{d_r}^T \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_{d_c}^T \otimes \mathbf{I}_{n_c}\right)
\end{aligned}
$$

where we apply the following property $\mathcal{A}\mathcal{B} \otimes \mathcal{C}\mathcal{D} = \left(\mathcal{A} \otimes \mathcal{C}\right)\left(\mathcal{B} \otimes \mathcal{D}\right)$ of the Kronecker product $\otimes$. In the next chapter we are going to use this equality to efficiently solve the super-resolution optimization problem.

2.4 Image gradient

As we have seen the purpose of the regularization term is to preserve some properties of the original image in the solution. Many of the properties that we are interested in can be expressed in terms of the image gradient. The gradient of a function of two variables $f(x, y)$ is defined as the two dimensional column vector of the partial derivatives with respect to x and y

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

We can compute approximations of the partial derivatives of a digital image using finite differences. We will use forward finite differences as described by the following formulas

$$\frac{\partial u}{\partial x}(i, j) = u(i + 1, j) - u(i, j)$$

$$\frac{\partial u}{\partial y}(i, j) = u(i, j + 1) - u(i, j)$$

where $u(i, j)$ refers to the pixel at row i and column j of the image u. Therefore the image gradient is composed of two images of the same size as the input image and whose pixels represent the difference with the adjacent pixel in the horizontal and vertical direction respectively. This is the same as applying the following filters through convolution

$$\frac{\partial u}{\partial x} = \begin{bmatrix} -1 & +1 \end{bmatrix} * u$$

$$\frac{\partial u}{\partial y} = \begin{bmatrix} -1 \\ +1 \end{bmatrix} * u$$

For this reason we can define the partial derivatives of $\mathbf{u} \in \mathbb{R}^N$ as the application of the linear operator $\mathbf{D}_h \in \mathbb{R}^{N \times N}$ and $\mathbf{D}_v \in \mathbb{R}^{N \times N}$ and the image gradient as $\mathbf{D} \in \mathbb{R}^{2N \times N}$ such that $\mathbf{Du} = (\mathbf{D}_h\mathbf{u}; \mathbf{D}_v\mathbf{u})$. Since $\mathbf{D}_h$ and $\mathbf{D}_v$ are BCCB matrices we can compute the image gradient efficiently in

the frequency domain in the same way as we did for the blurring operator. Therefore, there exist two diagonal matrices $\mathbf{\Sigma}_h \in \mathbb{R}^{N\times N}$ and $\mathbf{\Sigma}_v \in \mathbb{R}^{N\times N}$ such that

$$\mathbf{D}_h = \mathbf{F}^H \mathbf{\Sigma}_{\boldsymbol{h}} \mathbf{F} \quad \text{and} \quad \mathbf{D}_v = \mathbf{F}^H \mathbf{\Sigma}_{\boldsymbol{v}} \mathbf{F}.$$

where $\mathbf{F}$ and $\mathbf{F}^H$ are the two-dimensional discrete Fourier transform matrix and its inverse.

Chapter 3

Methods

3.1 ADMM algorithm

The ADMM (Alternating Direction Method of Multipliers) algorithm solves constrained optimization problems in the form

$$\begin{aligned} \text{minimize} \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned}$$

with variables $x \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$ where $A \in \mathbb{R}^{p \times n}$, $B \in \mathbb{R}^{p \times m}$ and $c \in \mathbb{R}^p$. This method can be used for single variable optimization problems where the variable x that we want to minimize is split in two parts, x and z in this case, and with objective function separated into f and g across this splitting.
The optimal value is denoted by

$$x^* = \inf \{ f(x) + g(z) \mid Ax + Bz = c \}$$

The augmented Lagrangian function is given by

$$L_\beta(x, z, \lambda) = f(x) + g(z) + \lambda^T (Ax + Bz - c) + \frac{\beta}{2} ||Ax + Bz - c||_2^2$$

and the general algorithm for ADMM consists of the following iterations

$$x^{k+1} = \arg\min_x L_\beta(x, z^k, \lambda^k) \tag{1}$$

$$z^{k+1} = \arg\min_x L_\beta(x^{k+1}, z, \lambda^k) \tag{2}$$

$$\lambda^{k+1} = \lambda^k + \beta(Ax^{k+1} + Bz^{k+1} - c) \tag{3}$$

where $\beta > 0$, è is the penalty parameter. The three steps of the algorithm are the minimization of x (1), the minimization of z (2) and the update of the dual variable λ using a step of length β (3). The variables x and z are updated in an alternating fashion which accounts for the name Alternating Direction.

We can also write the algorithm in a more convenient form combining the linear term $\lambda^T(Ax + Bz - c)$ and the quadratic term $\frac{\beta}{2}||Ax + Bz - c||_2^2$ of the augmented Lagrangian function L_β and by scaling the dual variable λ.
We define the value of the residual $r = Ax + Bz - c$ and obtain the following equalities

$$\begin{aligned}
\lambda^T r + \frac{\beta}{2}||r||_2^2 &= \lambda^T r + \frac{\beta}{2}||r||_2^2 + \frac{1}{2\beta}||\lambda||_2^2 - \frac{1}{2\beta}||\lambda||_2^2 \\
&= \frac{\beta}{2}(||r||_2^2 + \frac{2}{\beta}\lambda^T r + \frac{1}{\beta^2}||\lambda||_2^2) - \frac{1}{2\beta}||\lambda||_2^2 \\
&= \frac{\beta}{2}||r + \frac{\lambda}{\beta}||_2^2 - \frac{1}{2\beta}||\lambda||_2^2 \\
&= \frac{\beta}{2}||r + u||_2^2 - \frac{\beta}{2}||u||_2^2
\end{aligned}$$

where $u = \frac{\lambda}{\beta}$ is the scaled dual variable. Using this form we can rewrite the 3 passages of the algorithm in the following way

$$x^{k+1} = \arg\min_x f(x) + \frac{\beta}{2}||Ax + Bz^k - c + u^k||_2^2 \tag{1}$$

$$z^{k+1} = \arg\min_z g(z) + \frac{\beta}{2}||Ax^{k+1} + Bz - c + u^k||_2^2 \tag{2}$$

$$u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c \tag{3}$$

We can also define the residual at iteration k as $r^k = Ax^k + Bz^k - c$ and

compute the value u^k as the running sum of the residuals

$$u^{k+1} = u^k + r^{k+1} = u^0 + \sum_{j=1}^{k+1} r^j$$

This last form of the ADMM is known as the scaled form, since it's expressed in terms of the scaled dual variable. Although the two forms are equivalent, we are going to express the algorithms in the next section in the scaled form as the formulas are often shorter and simpler. We refer to [7] for more details on the ADMM algorithm.

3.2 ADMM for super-resolution

We can apply the ADMM algorithm to the super-resolution problem by setting the objective functions $f(\mathbf{u}) = \frac{1}{2}||\mathbf{SHu} - \mathbf{g}||_2^2$ which represents the fidelity term and $g(\mathbf{z}) = \mu R(\mathbf{z})$ which represents the regularization term with parameter μ. Regarding the constraint, we have that $A = \mathbf{D}$ is the linear application that computes the image gradient, $B = -I$ is the negated identity matrix and $c = 0$ the null vector. The super-resolution problem can then be rewritten as

$$\begin{aligned} \underset{\mathbf{u}\in\mathbb{R}^{N_h}}{\arg\min} \quad & \frac{1}{2}||\mathbf{SHu} - \mathbf{g}||_2^2 + \mu R(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{Du} - \mathbf{z} = 0 \end{aligned}$$

The augmented Lagrangian is given by

$$L_\beta(\mathbf{u}, \mathbf{z}, \boldsymbol{\lambda}) = \frac{1}{2}||\mathbf{SHu} - \mathbf{g}||_2^2 + \mu R(\mathbf{z}) + \boldsymbol{\lambda}^T(\mathbf{Du} - \mathbf{z}) + \frac{\beta}{2}||\mathbf{Du} - \mathbf{z}||_2^2$$

where $\beta > 0$ is the penalty parameter and $\boldsymbol{\lambda}$ is the dual variable, also known as the Lagrange multiplier. The three ADMM steps are given by the following

scheme

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \mu R(\mathbf{z}) + \frac{\beta}{2}||\mathbf{D}\mathbf{u}^k - (\mathbf{z} - \frac{\boldsymbol{\lambda}^k}{\beta^k})||_2^2 \tag{1}$$

$$\mathbf{u}^{k+1} = \arg\min_{\mathbf{u}} \frac{1}{2}||\mathbf{S}\mathbf{H}\mathbf{u} - \mathbf{g}||_2^2 + \frac{\beta}{2}||\mathbf{D}\mathbf{u} - (\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})||_2^2 \tag{2}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \beta^k(\mathbf{D}\mathbf{u}^{k+1} - \mathbf{z}^{k+1}) \tag{3}$$

We compute the solution by solving subproblems (1) and (2) and updating the dual variable $\boldsymbol{\lambda}$ (3) iteratively. In the next section we analyse an efficient solution for subproblem (2) and in the rest of the chapter we see two possible choices for the regularization term $R(\mathbf{z})$ and the respective solutions for subproblem (1).

3.3 Fast Super-Resolution

Our main contribution is the implementation of the following fast super-resolution algorithm as described in [2].
We show, under the hypothesis considered on the discrete operators $\mathbf{S}$ and $\mathbf{H}$ in the previous chapter how subproblem (2) can be efficiently solved in the frequency domain. The problem is a standard SR $\ell_2 - \ell_2$ optimization problem which admits a solution since the objective considered is convex. In particular, by applying the first order optimality conditions, a solution $\mathbf{u}^{k+1}$ of (2) can be viewed as the solution of the following linear system:

$$(\mathbf{H}^H\mathbf{S}^H\mathbf{S}\mathbf{H} + \beta^k\mathbf{D}^H\mathbf{D})\mathbf{u}^{k+1} = \left(\mathbf{H}^H\mathbf{S}^H\mathbf{g} + \beta^k\mathbf{D}^H(\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})\right).$$

In [1] the problem of finding $\mathbf{u}^{k+1}$ is addressed by applying the Conjugate Gradient (CG) iterative scheme. Despite its good performances, this approach is not efficient in terms of the overall computational cost, therefore a direct solver is preferable. Unfortunately, because of the particular structure of the decimation matrix, the joint operator $\mathbf{SH}$ cannot be diagonalized in the frequency domain, thus preventing any direct implementation of $\mathbf{u}^{k+1}$.

By considering the assumptions in the previous chapter and manipulating the last expression in terms of $\mathbf{F}$ and $\mathbf{F}^H$ we deduce the following chain of equalities:

$$
\begin{aligned}
(\mathbf{F}^H\mathbf{\Lambda}^H\mathbf{F}\mathbf{S}^H\mathbf{S}\mathbf{F}^H\mathbf{\Lambda}\mathbf{F} + \beta^k\mathbf{F}^H\left(\boldsymbol{\Sigma_h}^H\boldsymbol{\Sigma_h} + \boldsymbol{\Sigma_v}^H\boldsymbol{\Sigma_v}\right)\mathbf{F})\mathbf{u}_{k+1} &= \mathbf{r}^k \\
(\mathbf{F}^H\mathbf{\Lambda}^H\frac{1}{d}\left(\mathbf{J}_{d_r}\otimes\mathbf{I}_{n_r}\right)\otimes\left(\mathbf{J}_{d_c}\otimes\mathbf{I}_{n_c}\right)\mathbf{\Lambda}\mathbf{F} + \beta^k\mathbf{F}^H\left(\boldsymbol{\Sigma_h}^H\boldsymbol{\Sigma_h} + \boldsymbol{\Sigma_v}^H\boldsymbol{\Sigma_v}\right)\mathbf{F})\mathbf{u}_{k+1} &= \mathbf{r}^k \\
(\mathbf{\Lambda}^H\frac{1}{d}\left(\mathbf{J}_{d_r}\otimes\mathbf{I}_{n_r}\right)\otimes\left(\mathbf{J}_{d_c}\otimes\mathbf{I}_{n_c}\right)\mathbf{\Lambda}\mathbf{F} + \beta^k\left(\boldsymbol{\Sigma_h}^H\boldsymbol{\Sigma_h} + \boldsymbol{\Sigma_v}^H\boldsymbol{\Sigma_v}\right)\mathbf{F})\mathbf{u}_{k+1} &= \mathbf{F}\mathbf{r}^k \\
(\frac{1}{d}\underline{\mathbf{\Lambda}}^H\underline{\mathbf{\Lambda}} + \beta^k\boldsymbol{\Sigma_h}^H\boldsymbol{\Sigma_h} + \boldsymbol{\Sigma_v}^H\boldsymbol{\Sigma_v})\mathbf{F}\mathbf{u}_{k+1} &= \mathbf{F}\mathbf{r}^k
\end{aligned}
$$

where $\mathbf{r}^k := \mathbf{H}^H\mathbf{S}^H\mathbf{g} + \beta^k\mathbf{D}^H(\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})$ and $\underline{\mathbf{\Lambda}} := \left(\mathbf{1}_{d_r}^T\otimes\mathbf{I}_{n_r}\right)\otimes\left(\mathbf{1}_{d_c}^T\otimes\mathbf{I}_{n_c}\right)\mathbf{\Lambda}$.

Therefore we deduce:

$$\mathbf{u}_{k+1} = \mathbf{F}^H(\frac{1}{d}\underline{\mathbf{\Lambda}}^H\underline{\mathbf{\Lambda}} + \beta^k\boldsymbol{\Sigma_h}^H\boldsymbol{\Sigma_h} + \boldsymbol{\Sigma_v}^H\boldsymbol{\Sigma_v})^{-1}\mathbf{F}\mathbf{r}^k.$$

Using the Woodbury formula as in [2], the expression of $\mathbf{u}_{k+1}$ reads as in the following:

$$\mathbf{u}_{k+1} = \frac{1}{\beta}\mathbf{F}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k - \frac{1}{\beta}\mathbf{F}^H\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\left(\beta d\mathbf{I}_{N_l} + \underline{\mathbf{\Lambda}}\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\right)^{-1}\underline{\mathbf{\Lambda}}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k.$$

we observe that under the assumptions of cyclic boundary conditions the discrete Laplacian $\mathbf{D}^H\mathbf{D}$ is not invertible. As in [2] we add to the objectives in (2) a regularization term $\sigma_{\boldsymbol{\mathcal{D}}}\lambda\|\mathbf{u}\|_2^2$ defined in terms of a small constant $1 \gg \sigma_{\boldsymbol{\mathcal{D}}} > 0$.

Upon this assumption, we set $\mathbf{\Psi} := \left(\boldsymbol{\Sigma}_h^H\boldsymbol{\Sigma}_h + \boldsymbol{\Sigma}_v^H\boldsymbol{\Sigma}_v + \sigma_{\boldsymbol{\mathcal{D}}}\mathbf{I}_{N_h}\right)^{-1}$.

The steps required to solve the super-resolution subproblem are summarized in Algorithm 1.

Algorithm 1 – FSR

input: $\mathbf{g}, \mathbf{S}, \mathbf{H}, \mathbf{D}_h, \mathbf{D}_v, \mathbf{z}^{k+1}, \beta, d$

output: $\mathbf{u}^{k+1}$

Factorizations of matrices $\mathbf{H}, \mathbf{D}_h, \mathbf{D}_v$

$\mathbf{H} = \mathbf{F}^H \mathbf{\Lambda} \mathbf{F}$

$\mathbf{D}_h = \mathbf{F}^H \mathbf{\Sigma_h} \mathbf{F}$

$\mathbf{D}_v = \mathbf{F}^H \mathbf{\Sigma_v} \mathbf{F}$

Compute $\underline{\mathbf{\Lambda}}$ *and* $\mathbf{\Psi}$

$\underline{\mathbf{\Lambda}} \leftarrow \left(\mathbf{1}_{d_r}^T \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_{d_c}^T \otimes \mathbf{I}_{n_c}\right) \mathbf{\Lambda}$

$\mathbf{\Psi} \leftarrow \left(\mathbf{\Sigma}_h^H \mathbf{\Sigma}_h + \mathbf{\Sigma}_v^H \mathbf{\Sigma}_v + \sigma_{\boldsymbol{\mathcal{D}}} \mathbf{I}_{N_h}\right)^{-1}$

Compute solution of the linear system

$\mathbf{r}^k \leftarrow \mathbf{H}^H \mathbf{S}^H \mathbf{g} + \beta^k \mathbf{D}^H (\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})$

$\mathbf{u}^{k+1} \leftarrow \frac{1}{\beta} \mathbf{F}^H \mathbf{\Psi} \mathbf{F} \mathbf{r}^k - \frac{1}{\beta} \mathbf{F}^H \mathbf{\Psi} \underline{\mathbf{\Lambda}}^H \left(\beta d \mathbf{I}_{N_l} + \underline{\mathbf{\Lambda}} \mathbf{\Psi} \underline{\mathbf{\Lambda}}^H\right)^{-1} \underline{\mathbf{\Lambda}}^H \mathbf{\Psi} \mathbf{F} \mathbf{r}^k.$

3.4 ℓ_0 gradient minimization

We are going to define the regularization term using the ℓ_0 norm of the image gradient. This choice is known to have a strong ability of edge-preserving flattening. For this reason it is especially useful for achieving piecewise constant segmentations of the input image. The ℓ_0 norm of the image gradient (or ℓ_0 gradient in short) is defined as follows

$$||\mathbf{Du}||_0 := \#\left\{||(\mathbf{Du})_i||, i = 1, \ldots, N : (\mathbf{Du})_i \neq \mathbf{0}\right\}$$

which represents the number of elements of the image gradient that are not null. Intuitively the ℓ_0 gradient is the number of pixels in the image in which there is a discontinuity or a "jump" when compared to the neighbouring pixels.

The first model we are going to consider minimizes the value of the ℓ_0 gradient by solving the following optimization problem

$$\arg\min_{\mathbf{u} \in \mathbb{R}^{N_h}} \frac{1}{2} ||\mathbf{SHu} - \mathbf{g}||_2^2 + \mu ||\mathbf{Du}||_0$$

We now see how to solve subproblem (1) of the ADMM scheme with this choice of the regularization term. For each iterate k we denote $\mathbf{v}^k$ the vector $\mathbf{D}\mathbf{u}^k + \frac{\boldsymbol{\lambda}^k}{\beta^k}$. Due to decomposability of the ℓ_0 term we observe that the objective in (1) is separable. Therefore, a solution $\mathbf{z}^{k+1} \in \mathbb{R}^{2N_h}$ of the optimization problem (1) is computed solving for $i = 1 \dots N_h$ the 2D optimization problems of the following form:

$$\arg\min_{\mathbf{z}_i \in \mathbb{R}^2} \delta^k ||\mathbf{z}_i||_0 + ||\mathbf{z}_i - \mathbf{v}_i^k||_2^2,$$

where $\delta^k = \frac{2\mu}{\beta^k}$. As explained in [1], computing $\mathbf{z}_i^k$ corresponds to the proximal mapping of the 2D ℓ_0 term with parameter δ^k evaluated in $\mathbf{v}^k$, which is the 2D hard-thresholding operator. The necessary steps are summarized in algorithm 2.

Algorithm 2 – ℓ_0 gradient prox

input: $\mathbf{D}\mathbf{u}^k$, λ^k, β^k, μ

output: $\mathbf{z}^{k+1}$

1: $\mathbf{v}^k \leftarrow \mathbf{D}\mathbf{u}^k + \frac{\lambda^k}{\beta^k}$

2: **for** $\mathbf{v}_i^k \in \mathbf{v}^k$ **do**

3: $$\mathbf{z}_i^{k+1} \leftarrow \begin{cases} 0 & \text{if } ||\mathbf{v}_i^k||_2^2 < \frac{2\mu}{\beta^k} \\ \mathbf{v}_i^k & \text{otherwise} \end{cases}$$

4: **end for**

3.5 ℓ_0 gradient projection

The positive scalar parameter μ does not directly correspond to the degree of flatness of the result, therefore the users have to deal with an heuristic selection of a suitable value of this parameter. While it weights the contribution of the ℓ_2 and ℓ_0 terms, it lacks of a physical meaning. More precisely, the larger the magnitude of μ, the smaller the ℓ_0 gradient value of the solution, although an explicit relation between the magnitude of μ and the number

of jumps is unavailable, and thus users cannot directly specify the degree of flatness of the output image in advance. For this reason [3] suggests the following constrained model

$$\underset{\mathbf{u}\in\mathbb{R}^{N_h}}{\arg\min}\ \frac{1}{2}||\mathbf{SHu}-\mathbf{g}||_2^2 \quad \text{s.t.} \quad ||\mathbf{Du}||_0 \leq \alpha.$$

This constrained formulation introduces the positive integer α as upper bounds of the ℓ_0-gradient. In the unconstrained formulation described in the previous section, the parameter μ does not directly correspond to the degree of flatness of the solution, whereas in the constrained formulation one can determine α based on the information on the observed image $\mathbf{g}$, such as a certain percentage of total number of pixels we want to preserve or on the value ℓ_0-gradient in $\mathbf{g}$. Therefore, assigning a proper α is more intuitive and meaningful than looking for μ.

The constrained model can also be expressed as follows

$$\underset{\mathbf{u}\in\mathbb{R}^{N_h}}{\arg\min}\ \frac{1}{2}||\mathbf{SHu}-\mathbf{g}||_2^2 + i_{\{\|\cdot\|_0\leq\alpha\}}(\mathbf{z})$$

where by $i_{\{\|\cdot\|_0\leq\alpha\}}(\mathbf{z})$ we denote the indicator function of the non-convex set $\{\mathbf{z}\in\mathbb{R}^{2N_h} | \|\mathbf{z}\|_0 \leq \alpha\}$

$$i_{\{\|\cdot\|_0\leq\alpha\}}(\mathbf{z}) := \begin{cases} 0 & ||\mathbf{z}||_0 \leq \alpha \\ \infty & \text{otherwise} \end{cases}$$

As proposed in [3] a solution of subproblem (1) with this choice of the regularization term can be computed by the projection $\mathbf{z}^{k+1} = \mathbf{v}^k$ if and only if $\|\mathbf{v}^k\|_0 \leq \alpha$, otherwise $\mathbf{z}_i^{k+1} = \tilde{\mathbf{v}}_i^k$ for $i = 1 \dots N_h$, where

$$\tilde{\mathbf{v}}_i^k := \begin{cases} \mathbf{v}_i^k & i \in \{(1), \dots, (\alpha)\} \\ 0 & i \in \{(\alpha+1), \dots, (N_h)\} \end{cases}$$

and the indexes $(1), \dots (N_h)$ are computed by sorting in descending order the 2D subvectors $\mathbf{v}_i^k$ for $i = 1 \dots N_h$, in terms of their ℓ_2-norms and rela-

belling them accordingly. More simply, $\mathbf{z}^{k+1}$ is computed by replacing with zero the $N_h - \alpha$ elements of $\mathbf{v}^k$ with the smallest ℓ_2 norm.

Algorithm 3 – ℓ_0 gradient projection

input: $\mathbf{Du}^k$, λ^k, β^k, α

output: $\mathbf{z}^{k+1}$

1: $\mathbf{v}^k \leftarrow \mathbf{Du}^k + \frac{\lambda^k}{\beta^k}$

2: Compute $(1), \ldots, (N_h)$ by sorting the subvectors of $\mathbf{v}^k$ in descending order in terms of their ℓ_2 norm

3: **for** $\mathbf{v}_i^k \in \mathbf{v}^k$ **do**

4: $$\mathbf{z}_i^{k+1} \leftarrow \begin{cases} \mathbf{v}_i^k & i \in \{(1), \ldots, (\alpha)\} \\ 0 & i \in \{(\alpha+1), \ldots, (N_h)\}. \end{cases}$$

5: **end for**

Chapter 4

Results

In this chapter we are going to show the results of the ADMM algorithm for reconstruction based super-resolution that solves the optimization problem described in the previous chapters. We are going to compare the results of using the two regularization methods that we described, the unconstrained model, using the regularization parameter μ

$$\arg\min_{\mathbf{u}\in\mathbb{R}^{N_h}} \frac{1}{2}||\mathbf{SHu}-\mathbf{g}||_2^2 + \mu||\mathbf{Du}||_0$$

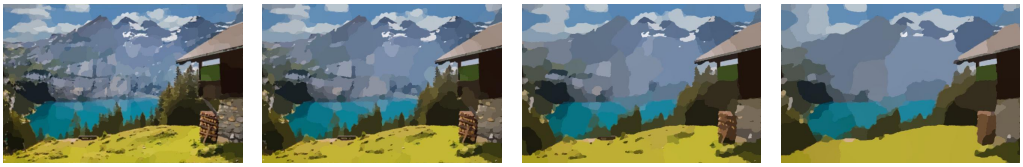
and the constrained model, where the desired value of the ℓ_0 gradient is specified by the parameter α

$$\arg\min_{\mathbf{u}\in\mathbb{R}^{N_h}} \frac{1}{2}||\mathbf{SHu}-\mathbf{g}||_2^2 \quad \text{s.t.} \quad ||\mathbf{Du}||_0 \leq \alpha.$$

We are going to test the algorithms by creating test problems from existing images; given a reference image (ground truth) we create a corrupted version through blurring, downsampling and the addition of noise. We will then evaluate the results of the algorithm on the corrupted image by comparing the reconstruction that we obtained with the ground truth and the input. The low resolution test images will be constructed using the decimation operator for downsampling with a factor of L, blurred with a Gaussian kernel of standard deviation σ_H and corrupted with additive Gaussian white noise of standard deviation σ_η.

One of the advantages of the constrained model over the unconstrained is the interpretation of the parameter α compared to that of the parameter

$\alpha = 25\%$ $\alpha = 17\%$ $\alpha = 10\%$ $\alpha = 7\%$



$\mu = 0.005$ $\mu = 0.01$ $\mu = 0.02$ $\mu = 0.04$

Figure 4.1: Examples of different choices of regularization parameter for the two methods. α is set to a percentage of the total number of pixels of the output image.

μ. Since α represents the value of the ℓ_0 gradient of the result, it's easier to estimate the level of segmentation that we are going to obtain. In the following examples we will compare the two algorithms by choosing values of the regularization parameter μ that result in similar values of the ℓ_0 gradient and thus a similar level of segmentation.

In figure 4.2 we highlight the effect of the regularization term based on the ℓ_0-norm of the image gradient. The graph below each image shows a line profile of the luminance of one of the rows. As expected, we can see that the graph is flatter when the value of the ℓ_0 gradient is lower, but hard edges in the original image are preserved.

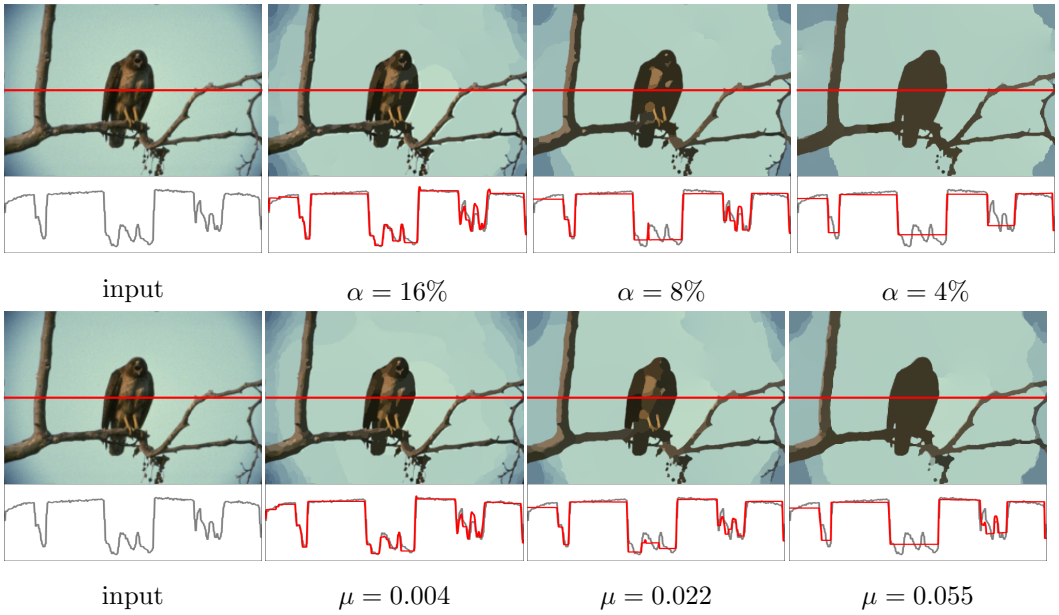


Figure 4.2: Luminance line profile for different values of the regularization parameter

4.1 Convergence

The regularization step for the constrained algorithm guarantees that the algorithm will converge to the desired value of the ℓ_0 gradient, therefore in our tests we use the following stopping criteria:

$$||\mathbf{Du}||_0 \leq 1.05 \cdot \alpha$$

whereas for the unconstrained method the algorithm is stopped when the relative change of the solution is lower than a fixed threshold, meaning that the image has not changed significantly in the last iteration:

$$\frac{||\mathbf{u}^{k+1} - \mathbf{u}^k||_2}{||\mathbf{u}^k||_2} \leq 5 \cdot 10^{-4}$$

In figure 4.3 we show a graph of the value of the ℓ_0 gradient at each iteration of the algorithm for different choices of the starting value. The three choices are a black image (zeros) an image obtained by scaling the input with a lanczos filter (lanczos) and an image obtained by smoothing the scaled input with a reconstruction algorithm based on the Total Variation

(TV). In all three cases we see that the value of the ℓ_0 gradient decreases to the desired value for the constrained model. For the unconstrained model the different starting points result in slightly different values of the ℓ_0 gradient.

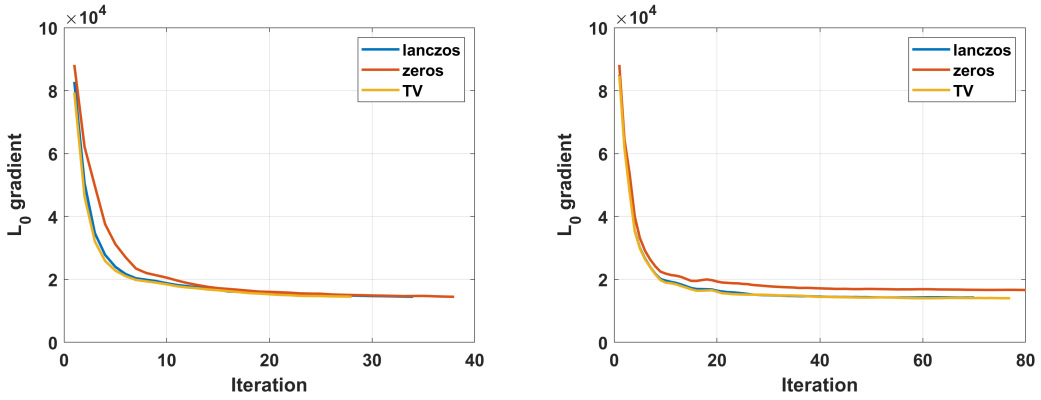


Figure 4.3: ℓ_0 gradient at each iteration for different choices of the starting value

The resulting images are shown in figure 4.4. We can see that due to the non-convexity of the regularization term the results are different based on the starting point, but since they have similar values of the ℓ_0 gradient they display a similar levels of segmentation.

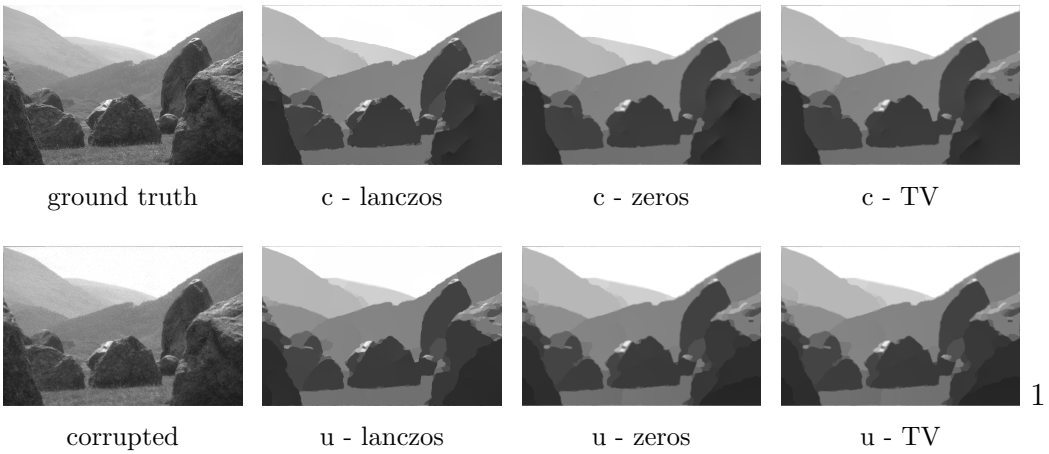


Figure 4.4: Results for different starting points. Top row: constrained with $\alpha = 9\%$. Bottom row: unconstrained with $\mu = 0.003$

4.2 Degradation

We now analyse the results of the algorithm on inputs created with different levels of degradation applied to the same reference image. In figure 4.5, 4.6 and 4.7 we see the results of varying the decimation factor, the standard deviation of the noise and the standard deviation of the Gaussian blur respectively.

Figure 4.5: From top to bottom: corrupted with a decimation factor of 2, 3, 4. From left to right: input, constrained, unconstrained

The values of the regularization parameters are fixed for each set of tests and they have been chosen so that the ℓ_0 gradient is approximately 6.5% of the total number of pixels. We see that the presence of noise in the input has very limited impact on the results, whereas we have loss of details in the reconstruction of images that are heavily blurred or downsampled.

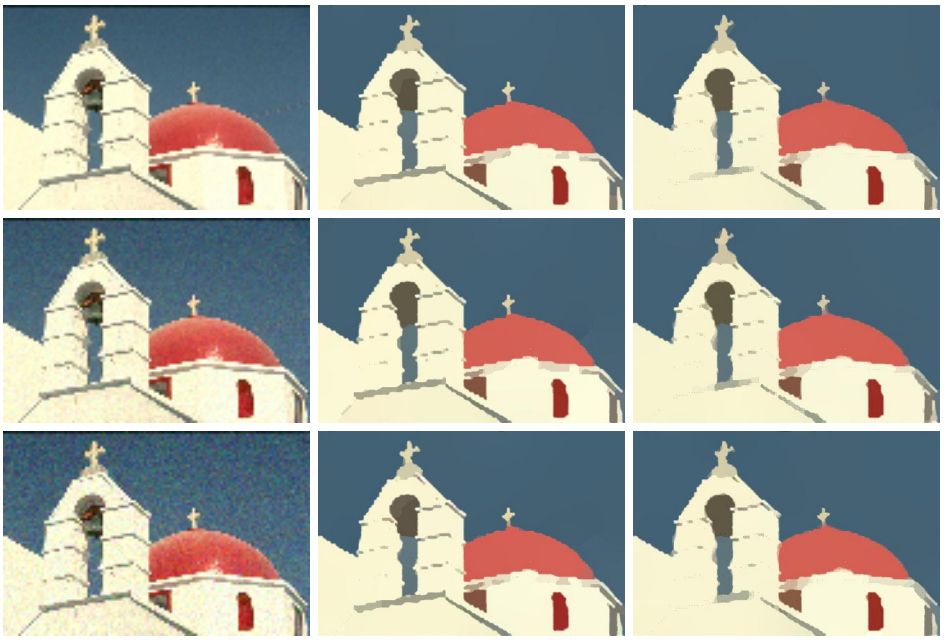


Figure 4.6: From top to bottom: corrupted with noise with standard deviation $\sigma_\eta = 0.1, 0.03, 0.05$. From left to right: input, constrained, unconstrained.

Figure 4.7: From top to bottom: corrupted with gaussian blur with standard deviation $\sigma_H = 0.5, 0.8, 1.0$. From left to right: input, constrained, unconstrained.

4.3 Time

The time complexity of the two algorithms is $O(kN \log N)$ where k is the number of ADMM iterations and N is the number of pixels in the high-resolution reconstruction. The time complexity of each iteration is given by the time required to solve the super-resolution subproblem and the regularization step. Since the super-resolution subproblem is solved in the frequency domain the most expensive operation is the computation of the Fourier transforms and its inverse, which requires time $O(N \log N)$. Furthermore the regularization subproblem has different time complexity depending on the chosen model. In the case of the constrained model we need to sort the elements of the image gradient in time $O(N \log N)$ whereas the unconstrained model is implemented with a thresholding operation which only requires time $O(N)$. In figure 4.8 see that the unconstrained model is faster in our tests, due to this difference.

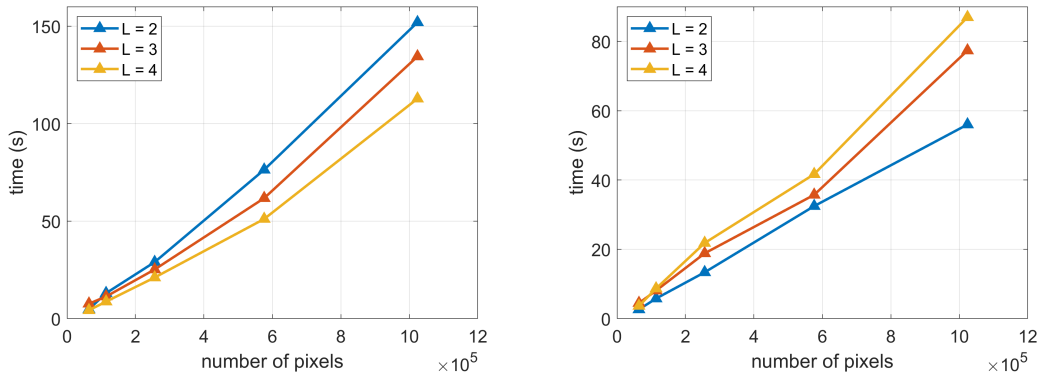


Figure 4.8: Execution time for color images of different sizes. The graph represents the total time over the number of pixels in the high resolution reconstruction.

Both models execute in 2 to 3 minutes for images of 10^6 pixels, making it feasible to run the algorithm on images of high dimensions or using high upscaling factors.

We now compare the execution time of the Conjugate Gradient (CG) iterative scheme used in [1] with the direct solver of our two algorithms (FSR). The closed form allows us to avoid the expensive iterations of the CG method, and the resulting algorithms are about 30 times faster in our tests.

	CG ($\mu = 0.003$)	FSR ($\mu = 0.003$)	FSR ($\alpha = 9\%$)
ADMM iterations	70	72	34
time per iteration	0.901 s	0.025 s	0.064 s
total time	63.11 s	1.79 s	2.19 s

Table 4.1: Execution time of the previous algorithm (CG) and our new method (FSR) on a black and white image of size 480×320

All the tests were executed in Matlab R2019b with an Intel i5-6500 and 8GB of RAM.

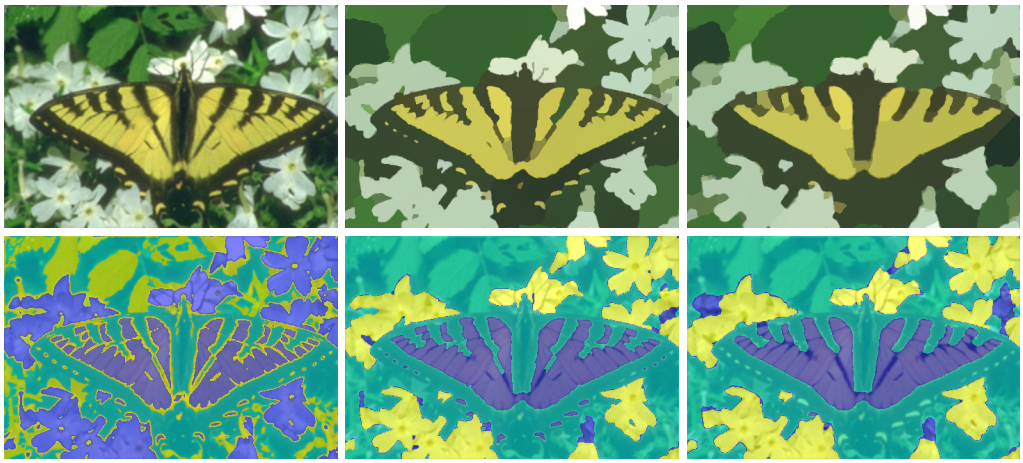
4.4 k-means segmentation

We have seen how the images produced by the algorithm show a good level of segmentation with an appropriate choice of the regularization parameter. We will now compare the result of a pixel labelling algorithm on the input images and on our reconstructions. The pixel labelling is created applying the k-means clustering algorithm in Matlab R2019b to the three color channels, which identifies a fixed number (k) of segments in the input image and assigns the index of the segment to each pixel.

Figure 4.9 and 4.10 show how applying this algorithm to an image that has been segmented with one of our methods results in a cleaner segmentation with more distinct boundaries between each segment.

input constrained unconstrained

Figure 4.9: k-means clustering image segmentation with $k = 4$. Top row: input to the k-means algorithm. Bottom row: pixel labelling on top of the input image



input constrained unconstrained

Figure 4.10: k-means clustering image segmentation with $k = 3$. Top row: input to the k-means algorithm. Bottom row: pixel labelling on top of the input image

Conclusions

We introduced reconstruction based methods for joint super-resolution and segmentation. Our main contribution is the implementation of a direct solution for the super-resolution subproblem as proposed in [2] to the joint super-resolution and segmentation approach in [1]. The new algorithm is about 30 times faster in our measurements. We have also compared the results of this algorithm with the constrained model proposed in [3]. The constrained and unconstrained models produce similar results at similar speed even when executed on heavily corrupted images.

Future research could explore different choices of the decimation of operator that result in higher quality super-resolution while maintaining the properties that allow for an efficient solution in the frequency domain. We briefly analysed the segmentation obtained from the algorithms but additional work is required to obtain higher quality pixel labelling from the reconstructions. As proposed in [1] the algorithms can be applied as a preprocessing step for Deep Learning based segmentation methods and could also be used to improve training and verification of neural networks for image segmentation.

The Matlab implementation of the algorithms and the code to generate all the images in the previous chapter are available at:
`https://github.com/ramenguy99/JointSuperResolutionSegmentation`.

Bibliography

[1] Pasquale Cascarano, Luca Calatroni, and Elena Loli piccolomini. Efficient ℓ_0 gradient-based super resolution for simplified image segmentation. *IEEE Transactions on Computational Imaging*, pages 1–1, 2021.

[2] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouamé, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for $\ell_2 - \ell_2$ problems. *IEEE Transactions on Image Processing*, 25(8):3683–3697, 2016.

[3] Shunsuke Ono. L_0 gradient projection. *IEEE Transactions on Image Processing*, 26(4):1554–1564, 2017.

[4] Martin Storath, Andreas Weinmann, Jürgen Frikel, and Michael Unser. Joint image reconstruction and segmentation using the potts model. *Inverse Problems*, 31(2):025003, 2015.

[5] Lukas Kiefer, Martin Storath, and Andreas Weinmann. PALMS Image Partitioning - A New Parallel Algorithm for the Piecewise Affine-Linear Mumford-Shah Model. *Image Processing On Line*, 10:124–149, 2020. `https://doi.org/10.5201/ipol.2020.295`.

[6] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via ℓ_0 gradient minimization. *ACM Trans. Graph (SIGGRAPH Asia)*, 2011.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the al-

ternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[8] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey, 2020.

[9] P.C. Hansen, J.G. Nagy, and D.P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics, 2006.

[10] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Prentice Hall, Upper Saddle River, N.J., 2008.