Darmstadt University of Technology
Institute of Telecommunications
Signal Processing Group
Prof. Dr.-Ing. A. M. Zoubir
Merckstraße 25,
64283 Darmstadt, Germany

Panasonic R&D Center Germany
Panasonic Frankfurt Laboratory
Monzastrasse 4c
63225 Langen

Panasonic

Master Thesis
Masterarbeit

High-Precision Motion Estimation for Video Sequences Considering Disturbing Factors

Hochpräzise Bewegungsschätzung für Videosequenzen unter Berücksichtigung von Störfaktoren

by / von
Christian Debes
29.09.2006

Supervisor / Betreuer : Prof. Dr. Abdelhak M. Zoubir
Dr. Chris L. Brown
Sven Dueking

Contents

List of Figures

List of Tables

Abstract

Classical motion estimators based on block similarity show a severe decrease in performance when aliasing and noise are present. A new scheme for high-precision motion estimation will be presented in this thesis, using an observation model for the image acquisition process. A Maximum Likelihood and Maximum A Posteriori (MAP) optimisation is performed to estimate the vector field between images. Different priors for stochastic regularisation are treated in the Maximum A Posteriori optimisation using Markov Random Field image models. The new motion estimator is compared to classical motion estimators in terms of the quality of the estimated vectorfield and computational complexity.

Notation Table

$b_{1,k}$	The k-th block from an image of interest for which a motion vector is searched, $k = 0, ..., K-1$
$b_{0,l}$	The l-th block out of the search range in a reference image, $l = 0, ..., L-1$
$b_{0,\tilde{l}}$	The matching block for $b_{1,k}$
$b_{1,k}(q)$	The q-th pixel from block $b_{1,k}$, $q = 0, ..., Q^2-1$
$B_{1,k}$	DCT transform of $b_{1,k}$
$B_{0,l}$	DCT transform of $b_{0,l}$
Q	The height and width of a quadratic image block
$C(b_{1,k}, b_{0,l})$	A general cost function for two image blocks
p	The increase factor for image interpolation
N_1	The number of vertical pixels in an image
N_2	The number of horizontal pixels in an image
$X(i,j)$	The (i,j)-th element of an image X in matrix notation
X_S	A shifted version of image X
X_L	A downsampled version of X_S
X_N	A noisy version of X_L
V	The true motion vector field
$\hat{V}$	The estimated motion vector field
$V(k) = (d^i(k), d^j(k))$	The true motion vector for the k-th block
$\hat{V}(k) = (\hat{d}^i(k), \hat{d}^j(k))$	The estimated motion vector for the k-th block
(d^i_G, d^j_G)	A global motion vector
s	The search range in the reference image
y_i	The i-th LR image in lexicographic notation, $i = 0, ..., F-1$
D	The downsampling matrix
B_i	The i-th blurring matrix
M_i	The i-th motion matrix
N	The neighbourhood matrix
x	The true underlying HR image in lexicographic notation
$x(n)$	The n-th pixel of x
η_i	The i-th noise vector in lexicographic notation
W_i	i-th noise covariance matrix
x_S	A shifted version of x
x_{SB}	A shifted and blurred version of x
$\hat{x}_B$	Bayesian estimate of x
$\hat{x}_{ML}$	Maximum Likelihood estimate of x
$\hat{x}_{MAP,G}$	Maximum A Posteriori estimate of x, Gaussian prior
$\hat{x}_{MAP,D}$	Maximum A Posteriori estimate of x, DE prior

σ_G	Scale parameter for the GMRF image model
σ_D	Scale parameter for the DEMRF image model
c	A general regularisation parameter
c_G	Regularisation parameter for the GMRF image model
c_D	Regularisation parameter for the DEMRF image model
$F_{ML}(x)$	Cost function for the ML approach
$F_{MAP,G}(x)$	Cost function for the MAP approach, Gaussian prior
$F_{MAP,D}(x)$	Cost function for the MAP approach, DE prior
M^j	The j-th motion matrix, $j = 1, ..., J$
$\hat{x}^r$	Estimate of x in the r-th iteration step
$\hat{g}^r$	Gradient at r-th iteration
$\hat{e}^r$	The conjugate direction at r-th iteration
R	Maximum number of iterations for the CG algorithm
ε	Threshold parameter for the CG algorithm
B_{max}	The maximal pixel intensity value

Chapter 1

Introduction

Motion analysis is an important research topic in image and video processing as well as in computer vision. Understanding how objects move is a field of high practical interest, covering a broad range of applications in many engineering fields. This introduction motivates the need for high-precise motion analysis and mentions some popular applications.
An image is assumed to be a projection of a 3D scene onto a 2D grid. A subsequent image in a video sequence usually represents partly the same physical region at another time. E.g. considering a camera which takes 25 frames per second, the time difference between two subsequent images is 40 ms. The difference between these two images is caused by a change of the physical region and/or the camera itself within the time difference of e.g. 40 ms. These changes include

- Motion of objects within the scene
- Motion of the camera
- Lighting changes
- Occlusions.

Furthermore due to a nonideal camera model, disturbing factors such as

- Charge-coupled device (CCD) sensor noise
- Blurring effects
- Aliasing effects

occur during the projection of the 3D scene onto a 2D grid. It is the aim of motion analysis to extract motion information caused by objects in the scene and by the camera from a sequence of images.

Extracting motion information from video sequences is a field of high practical importance, including applications such as

- Video tracking: Motion estimation is a crucial part for object tracking in video sequences with applications in remote video surveillance [4] and robot vision [12].
- Video compression: Knowledge of the motion between two images helps in removing temporal redundancy, thus leading to higher compression [19].
- Medical engineering: Applications for motion estimation can be found in numerous fields in medical engineering, e.g. patient positioning [1], motion compensation for image substraction in digital subtraction angiography (DSA) [8], study of face movement for psychiatric diagnoses [14] and many more.
- Super-Resolution Image Reconstruction: Super-Resolution Image Reconstruction provides methods to combine spatially displaced low resolution images into an image of higher resolution by making use of aliasing effects. Computing motion information between the low resolution images is the first step in Super-Resolution. High-Precision motion estimation is the most important part for a successful image reconstruction [18].

Classical motion estimators are based on the assumption of intensity conservation [2]. However this assumption may not be fulfilled in practice, e.g. in the presence of aliasing and noise. This results in unreliable motion estimates leading to problems in the applications mentioned above which all require an accurate estimation of the motion vectorfield as a first step.
This thesis is structured as follows:

- Chapter 2 explains the concept of fractional block based motion estimation and provides simulation results showing the performance of classical motion estimators.
- In Chapter 3 the concept of Super-Resolution is treated, explaining the multisensor concept for images and the stochastic regularisation approach used for image reconstruction.
- In Chapter 4 the Super-Resolution based motion estimator is presented, a new approach for motion estimation based on a multisensor approach.
- Chapter 5 gives a conclusion about the work done in this thesis and provides an outlook about future work which may be done in this area.

Chapter 2

Motion Analysis

This chapter provides an overview about motion analysis in video sequences. Section 2.1 explains the concept of blockbased motion estimation in detail, section 2.2 explains the simulation scheme and quality measures for motion estimation whereas section 2.3 shows simulation results of different motion estimation schemes when disturbing factors are present. Finally, conclusions are given in section 2.4.

2.1 Block-based motion estimation

One of the fundamental assumptions in motion estimation is the concept of intensity conservation [2] in which pixel intensities are assumed to be constant within a tracking period. If we furthermore assume a short time distance between two subsequent images of a video sequence, a translational motion model for small blocks of the image is sufficient to describe the spatial displacement.
Throughout this thesis we will restrict ourselves to matching-based measurements of motion, segmenting a luminance image into blocks and searching for matching blocks in one or more reference images. The idea of image division into blocks is visualised in figure 2.1. For reasons of brevity we assume all blocks $b_{1,0}, ..., b_{1,K-1}$ to have equal quadratic size $Q \times Q$ and the image to be divided into non-overlapping blocks as shown in figure 2.1. However, all concepts in this chapter are easily extendable to overlapping blocks of arbitrary shapes.
Making the above-mentioned assumptions of intensity conservation and translational motion a block for which motion information is needed may be compared to a set of overlapping blocks within a reference image. One may then define a distance measure and choose the reference block which matches best as being the block which describes the same physical region of the scene. The blockmatching principle is visualised in figure 2.2. Figure 2.2 displays a block $b_{1,k}$ from an image of interest for which a matching block $b_{0,\tilde{l}}$ with $0 \leq \tilde{l} \leq L-1$ and thus a motion vector describing the motion between $b_{1,k}$ and $b_{0,\tilde{l}}$ is wanted. $b_{1,k}$ is compared to a set of blocks $b_{0,l}$ within

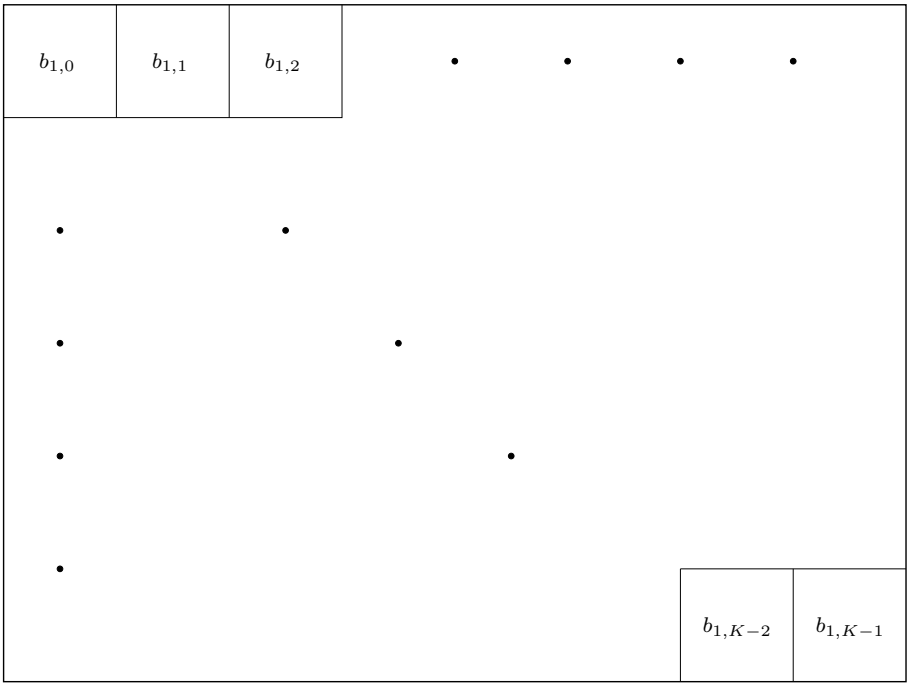


Figure 2.1: Image division into blocks

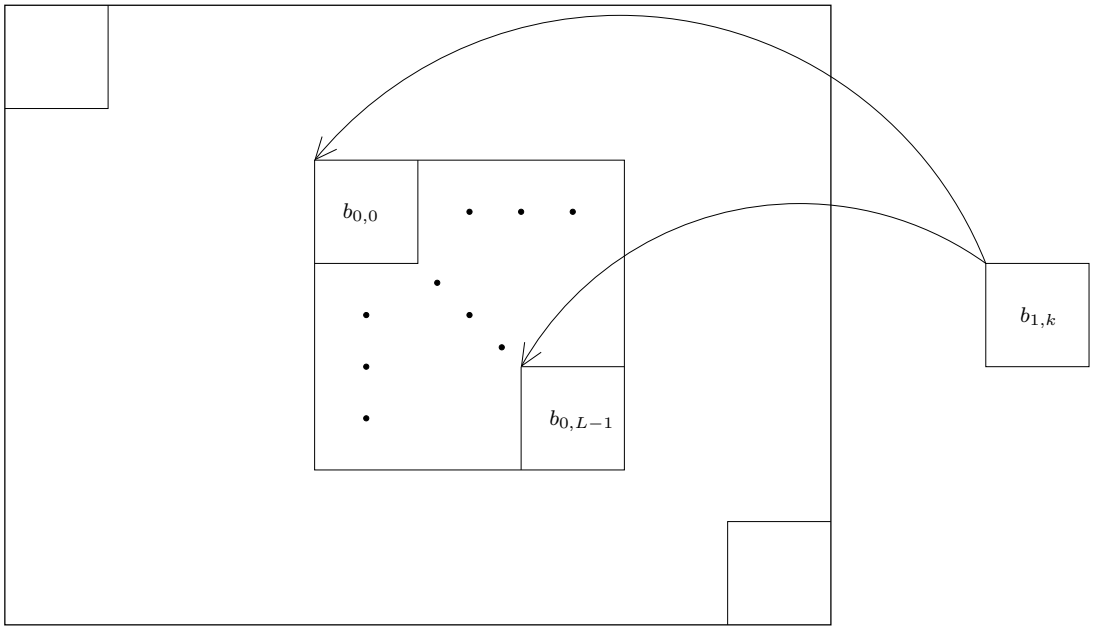


Figure 2.2: The blockmatching principle

a region $0 \leq l \leq L-1$ in the reference image.

2.1.1 Distance measures

Recalling the assumption of intensity conservation, several distance measures for block-comparison have been established within the last decades. We will restrict ourselves to the Sum of Absolute Difference (SAD), the Sum of Squared Differences (SSD), the Sum of Absolute Transform Differences (SATD) and the Sum of Squared Transform Differences (SSTD). Let $b_{0,l}(q)$ and $b_{1,k}(q)$ be the q-th pixel value of block $b_{0,l}$ and $b_{1,k}$ respectively, $q = 0, ..., Q^2 - 1$. The SAD and SSD distance measure are then defined as

$$\mathrm{SAD}(b_{1,k}, b_{0,l}) = \sum_{q=0}^{Q^2-1} |b_{1,k}(q) - b_{0,l}(q)| \tag{2.1}$$

and

$$\mathrm{SSD}(b_{1,k}, b_{0,l}) = \sum_{q=0}^{Q^2-1} (b_{1,k}(q) - b_{0,l}(q))^2 \tag{2.2}$$

giving a measure about how close $b_{1,k}$ and $b_{0,l}$ are in the pixel intensity-domain.
Let $B_{0,l} = T(b_{0,l})$ and $B_{1,k} = T(b_{1,k})$ be representations of $b_{0,l}$ and $b_{1,k}$ in the frequency domain. As one of the most frequently used transformations in image processing, the discrete cosine transform (DCT) [17] will be used in the following, leading to blocks $B_{0,l} = DCT(b_{0,l})$ and $B_{1,k} = DCT(b_{1,k})$ of the same size $Q \times Q$. The DCT is known to provide a strong energy compaction in the sense that it approaches the Karhunen-Loève-transform [9]. The SATD and SSTD using the DCT are defined as

$$\mathrm{SATD}(b_{1,k}, b_{0,l}) = \sum_{q=0}^{Q^2-1} |B_{1,k}(q) - B_{0,l}(q)| \tag{2.3}$$

and

$$\mathrm{SSTD}(b_{1,k}, b_{0,l}) = \sum_{q=0}^{Q^2-1} (B_{1,k}(q) - B_{0,l}(q))^2 \tag{2.4}$$

giving a measure about how close $b_{1,k}$ and $b_{0,l}$ are in the frequency-domain.
The principle of blockmatching will be shown within a small example. In figure 2.3(a) a 16×16 block is shown. The corresponding block and, thus, a motion vector is to be searched for within a search area in a reference image. The search area in this example is of size 24×24 (figure 2.3(b)). The block is now compared to all possible $(24 - 16 + 1)^2 = 81$ reference blocks in the search area which can be seen in figure 2.4. The SAD and SATD cost function, dependent on the position in vertical and horizontal direction in the search area (which are denoted as i and j respectively) are visualised in figure 2.5 and 2.6. It can clearly be seen that the SAD and SATD cost functions reach their minimum at the matching position $(i, j) = (5, 5)$. Several algorithms for finding the minimum of the cost functions are available, managing

the tradeoff between accuracy and computation time. In this thesis we will restrict ourselves to the most accurate and computationally intensive full-search algorithm (FSA). Given a general cost function $C(b_{1,k}, b_{0,l})$ the FSA works as follows: $b_{1,k}$ is compared to all possible blocks $b_{0,l}$, $0 \leq l \leq L-1$ in the search area of the reference image. The matching block $b_{0,\tilde{l}}$ is the calculated as $\tilde{l} = \arg\min_l \{C(b_{1,k}, b_{0,l})\}$. In applications where computational power is limited or real-time constraints are given fast search algorithms such as the Three Step Search (TSS) or the Nearest Neighbour Search (NNS) are preferred. For details see [5].

(a) Current block (b) Search area in the reference image

Figure 2.3: Blockmatching example

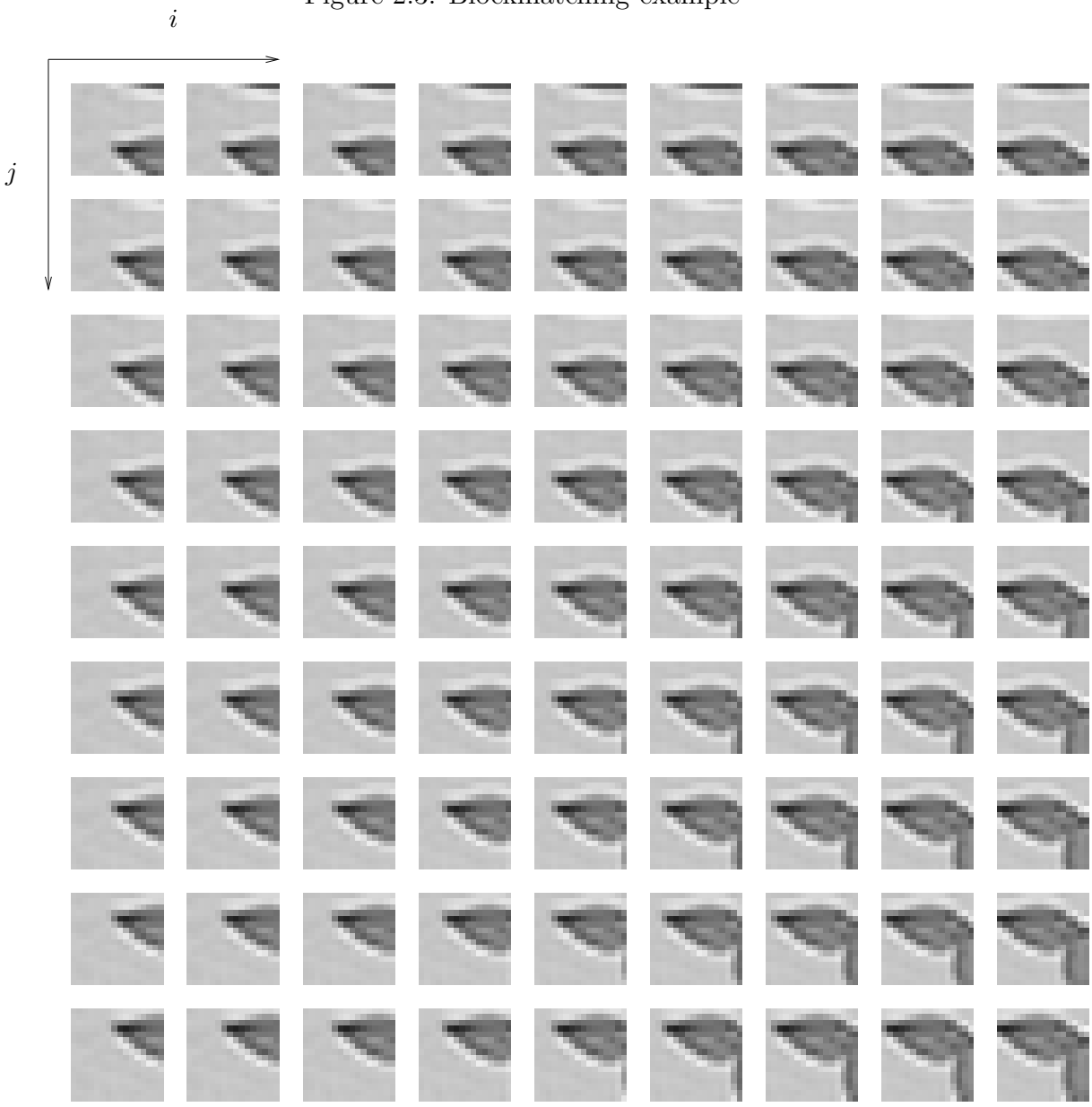


Figure 2.4: Blockmatching example: 81 reference blocks in the search area

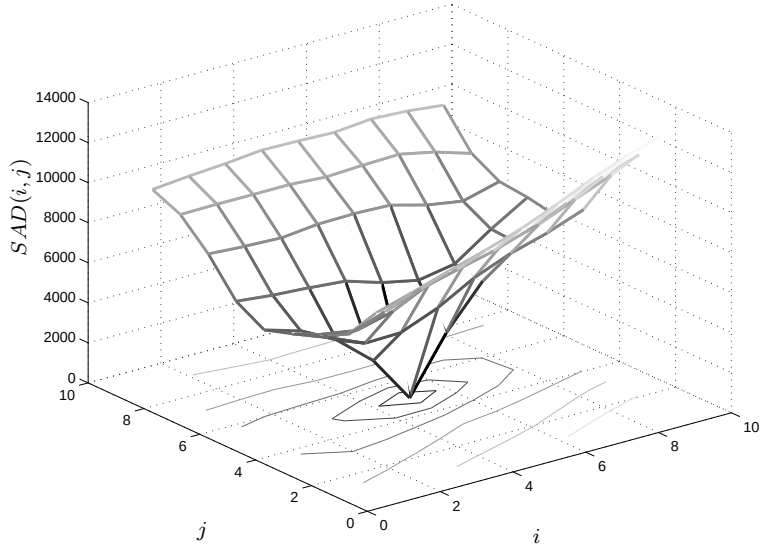


Figure 2.5: Blockmatching example: SAD cost function

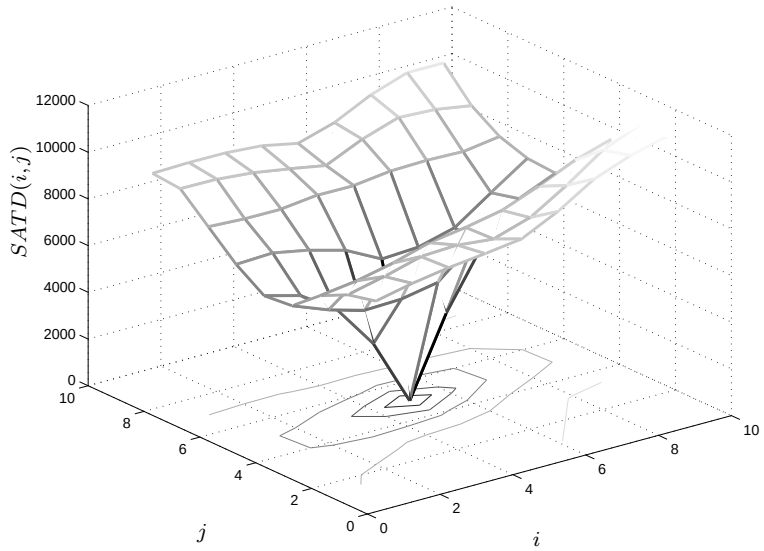


Figure 2.6: Blockmatching example: SATD cost function

2.1.2 Fractional-block-based motion estimation

Up to now we restricted ourselves to motion estimates with a resolution of integer pixel values. In order to increase the motion vector resolution to $\frac{1}{p}$ pixels it is a common technique to increase the image size by a factor of p in horizontal and vertical direction by using an interpolation scheme. This technique is visualised in figure 2.7.

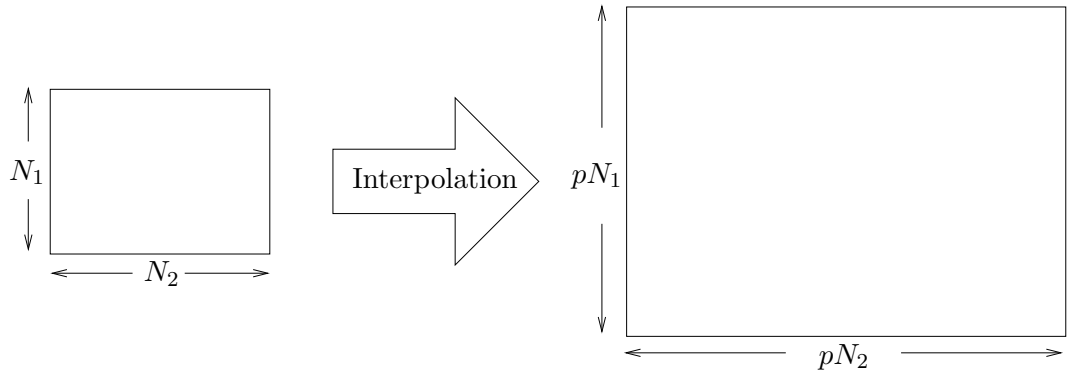


Figure 2.7: Image interpolation

Within this thesis three basic interpolation techniques will be treated, namely

- The nearest neighbour interpolation [6], the most simple interpolation scheme. New values are calculated by selecting the value of the nearest point without considering any other neighbours.
- The bilinear interpolation [6]. Based on the 4 nearest neighbours a linear interpolation in both dimensions is performed to calculate the new value.
- The bicubic interpolation [6]. The bicubic interpolation is frequently used for image scaling. It consideres the 16 nearest neighbours and calculates new pixel values using cubic interpolation polynomials.

Furthermore we will treat a 6 tap FIR interpolation filter which is currently used in a range of applications such as video coding [19] and image reconstruction [6]. Its weights for $p = 2$ are:

$$h_{6tap} = \left[\frac{1}{32}, -\frac{5}{32}, \frac{20}{32}, \frac{20}{32}, -\frac{5}{32}, \frac{1}{32}\right] \tag{2.5}$$

This filter is approximating an ideal interpolation filter, following the sinc function. In figure 2.8 the interpolation scheme using the 6 tap filter is shown for horizontal interpolation. Given six neighbouring samples in one image line (schematised as filled circles) a new interpolated sample can be generated by applying the weights from equation 2.5 to the image samples and adding them together. The result of

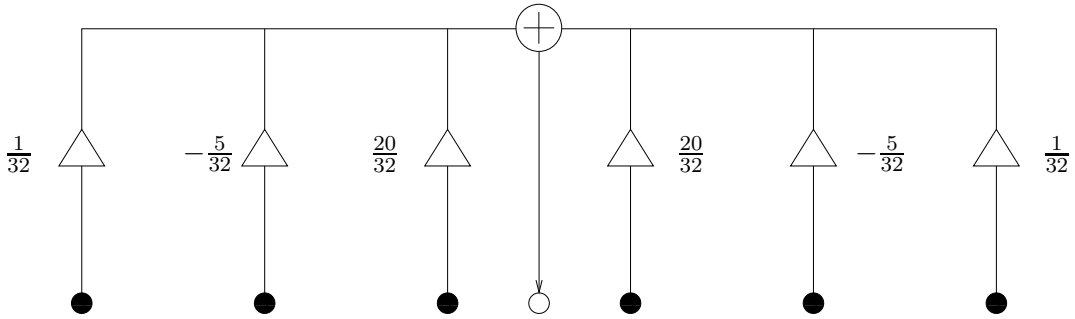


Figure 2.8: The 6 tap FIR filter interpolation scheme

applying the different interpolation filters to an image is visualised for $p = 2$ in the following: Figure 2.9(a) shows a part of the first frame of the 'Mobile' sequence, the interpolated versions are shown in figure 2.9(b)-(e).

(a) Original image

(b) Nearest neighbour interpolation

(c) Bilinear interpolation

(d) Bicubic interpolation

(e) 6 tap filter interpolation

Figure 2.9: Example: Interpolation methods

2.2 Simulation scheme

In order to quantitatively evaluate the performance of a motion estimator, a simulation study is necessary. This section describes in detail how motion between two images is simulated and how the quality of an estimated vectorfield is calculated.

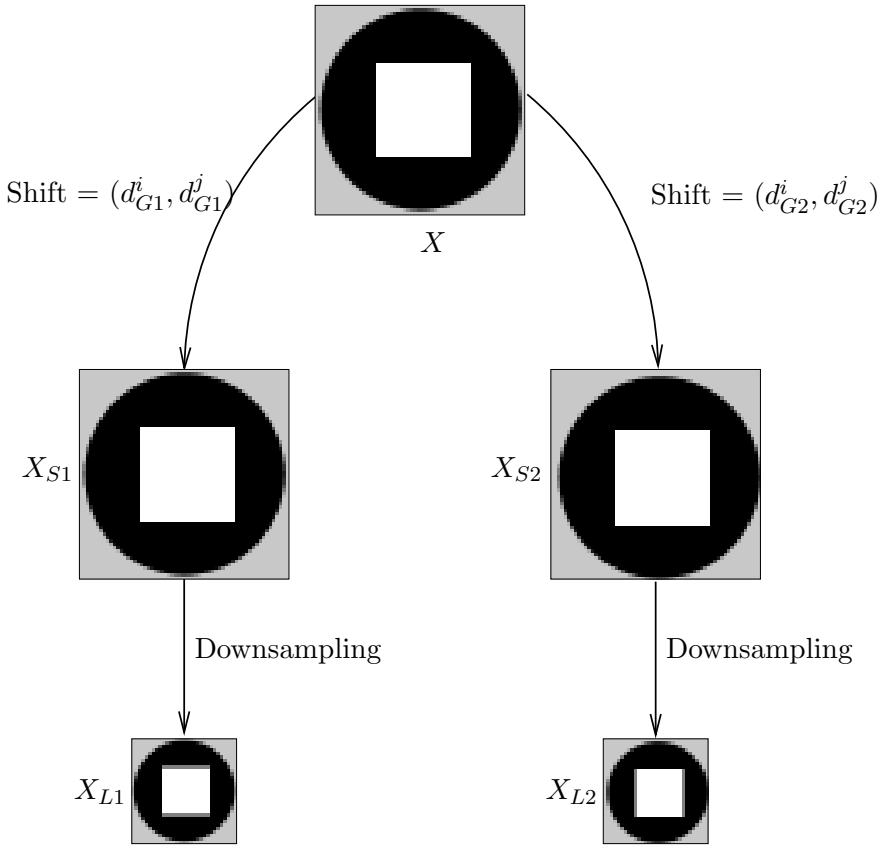


Figure 2.10: Simulation setup for motion estimation

2.2.1 Simulation scheme for motion estimation

The setup to simulate fractional pixel displacements is shown in figure 2.10. Given an image X of resolution $N_2 \times N_1$ with $X(i,j)$ being its elements, $i = 0, ..., N_2 - 1$ and $j = 0, ..., N_1 - 1$, a spatially displaced version of it can be constructed by shifting the original image globally by d^i_{G1} in the horizontal and d^j_{G1} in the vertical direction on the image grid leading to an image X_{S1} . The motion vector representing the spatial displacement will be denoted as (d^i_{G1}, d^j_{G1}). The resulting image X_{S1} is given by

$$X_{S1}(i,j) = \begin{cases} X(i - d^i_{G1}, j - d^j_{G1}) & \text{for } d^i_{G1} \leq i < N_2 \vee d^j_{G1} \leq j < N_1 \\ X(i - d^i_{G1}, 0) & \text{for } j < d^j_{G1} \vee d^i_{G1} \leq i < N_2 \\ X(0, j - d^j_{G1}) & \text{for } i < d^i_{G1} \vee d^j_{G1} \leq j < N_1 \\ X(0,0) & \text{else} \end{cases} \tag{2.6}$$

A second image X_{S2} is now constructed by applying a shift of (d^i_{G2}, d^j_{G2}) to the original image, leading to two images with a relative displacement $d_{rel} = (d^i_{G2} - d^i_{G1}, d^j_{G2} - d^j_{G1})$. Without loss of generality we will set $d^i_{G1} = d^j_{G1} = 0$ and thus $X_{S1} = X$ and $d_{rel} = (d^i_{G2}, d^j_{G2}) := (d^i_G, d^j_G)$. The images X_{S1} and X_{S2} are now downsampled by a factor of p, replacing every $p \times p$ block with its average value. The resulting images X_{L1} and X_{L2} are of size $\frac{N_2}{p} \times \frac{N_1}{p}$ and have a relative fractional spatial displacement of $(\frac{d^i_G}{p}, \frac{d^j_G}{p})$. Note that due to a nonideal downsampling aliasing effects occur.
Let now X_{L1} be the reference image and X_{L2} be the current image for which a vectorfield is desired. Recalling the description of (fractional) block-based motion estimation X_{L2} may now be divided into blocks for which a matching block is searched for within a search area of X_{L1}.

2.2.2 Quality measure for motion estimation

The outcome of motion estimation is an estimated vectorfield $\hat{V}$ with $\hat{V}(k) \in \mathbb{R}^2$ being the estimated motion vector for the k-th block, $k = 0, ..., K-1$ with K being the number of all blocks in the image of interest. The estimated vector components for the k-th block in i- and j-direction are then denoted as $\hat{V}(k) = (\hat{d}^i(k), \hat{d}^j(k))$. Given the true vectorfield V with $V(k) = (d^i(k), d^j(k))$ being the true vector for the k-th block, the mean absolute deviation will be used as a quality measure for vectorfields in this thesis:

$$\mathrm{MAD}(V, \hat{V}) = \frac{1}{K} \sum_{k=0}^{K-1} \sqrt{(d^i(k) - \hat{d}^i(k))^2 + (d^j(k) - \hat{d}^j(k))^2} \tag{2.7}$$

The absolute deviation of two vectors is visualised in figure 2.11. The absolute deviation of an estimated motion vector $(\hat{d}^i(k), \hat{d}^j(k))$ to the true motion vector $(d^i(k), d^j(k))$ is the Euclidean distance $\sqrt{(d^i(1) - \hat{d}^i(1))^2 + (d^j(1) - \hat{d}^j(1))^2}$ which is a natural measure about how close two vectors are on an image plane.

2.3 Simulation results

In this section the performance of the above-mentioned block-based motion estimators is shown. For the simulation setup the luminance part of the first frames of the following YUV sequences will be used:

- **Coastguard**, 352×288 pixels
- **Mobile**, 352×288 pixels

The following parameters will be used for simulating the downsampling process

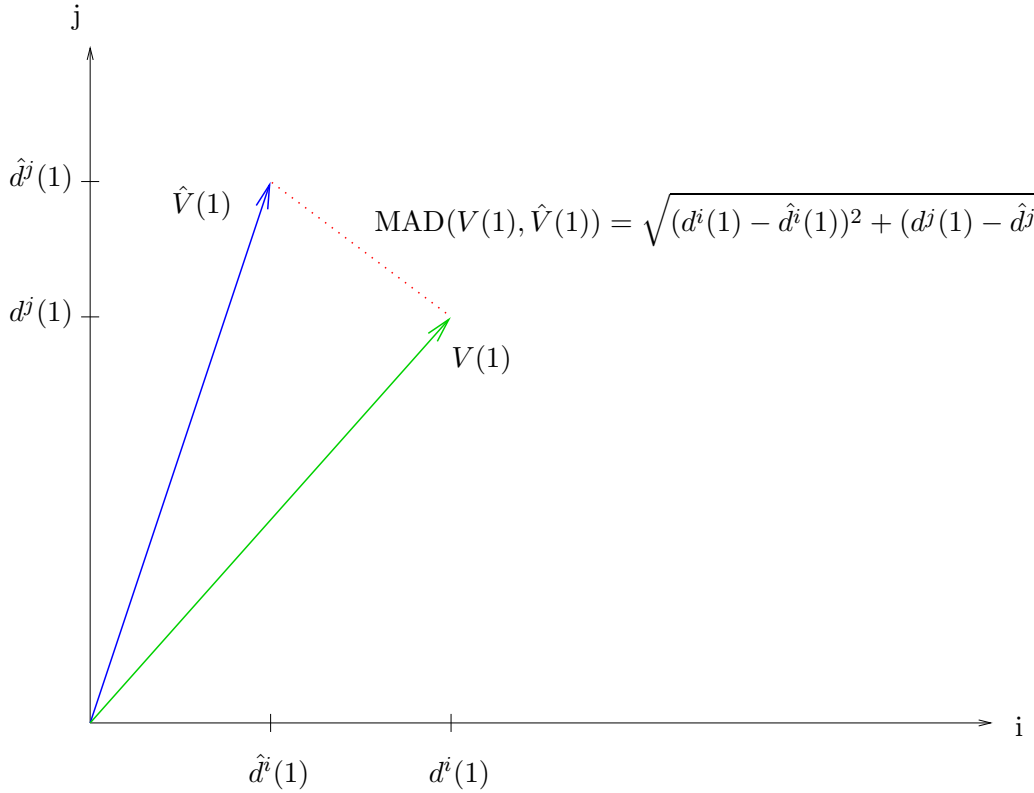


Figure 2.11: Absolute deviation of two vectors

- **Downsampling factor:** $p = 2$
- **Shift:** $(\frac{1}{2}, \frac{1}{2})$

Furthermore the motion estimator may take one of the following options:

- **Cost function:** $C \in \{\text{SAD; SSD; SATD; SSTD}\}$
- **Interpolation:** $IP \in \{\text{'Nearest neighbour'; 'Bilinear'; 'Bicubic'; '6 tap FIR'}\}$

The search range is fixed to $s = 2p$, yielding a window size of $(2p+1) \times (2p+1)$ in the reference image. The blocksize is set to 4×4.
By applying a nonideal downsampling filter and due to relative subpixel shifts in the resulting images X_{L1} and X_{L2} aliasing effects occur, violating the assumption of intensity conservation. Furthermore CCD sensor noise will be modeled as additive white Gaussian noise, resulting in noisy, aliased images X_N:

$$X_N = X_L + \eta \tag{2.8}$$

with $\eta \sim \mathcal{N}(0, \sigma_\eta^2)$. The simulation results are averaged over 100 Monte Carlo simulations with $\sigma_\eta = 0, 1, ...5$.
In the following we will show representative plots and vectorfields of some selected simulations. The complete simulation results of all test scenarios can be found in the appendix.

2.3.1 The effect of the interpolation scheme on motion estimates

In order to achieve halfpel accuracy each image involved in the motion estimation process has to be upsampled by a factor of $p = 2$. In figures 2.12 and 2.13 the effect of different interpolation schemes on the quality of the estimated vectorfield can be seen for the 'Coastguard' and 'Mobile' sequences. The nearest neighbour interpolation as being the most basic upsampling scheme offers the worst performance whereas the MAD decreases when more advanced interpolation schemes are used. It can furthermore be seen that the SSD and SSTD cost functions performs slightly better than the SAD and SATD cost function, however the difference vanishes when an advanced interpolation scheme such as the 6 tap FIR interpolation filter is used.

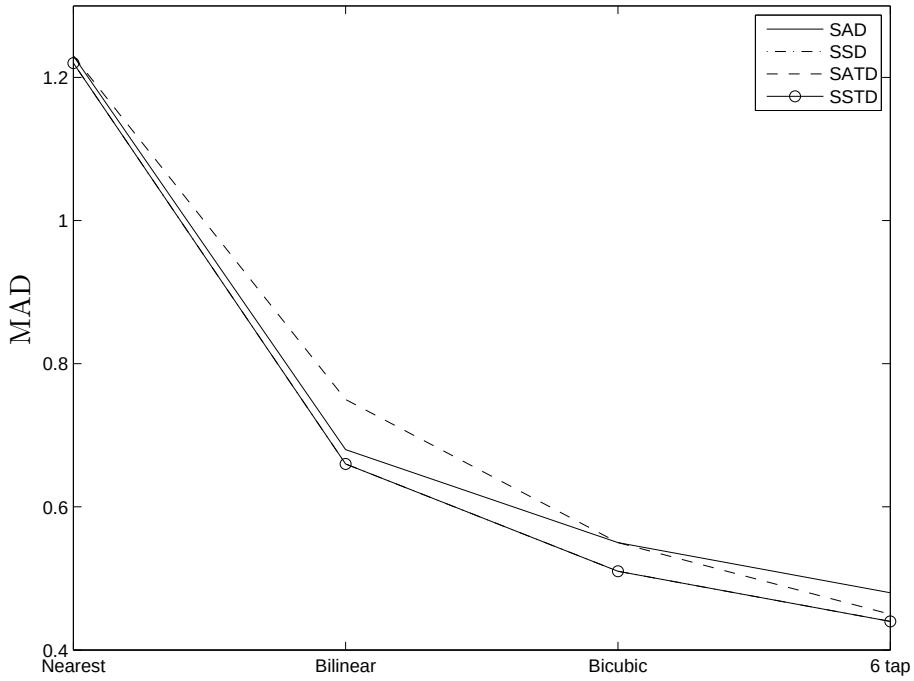


Figure 2.12: Simulation result: Effect of the interpolation scheme on motion estimation, 'Coastguard' sequence

2.3.2 The effect of noise on motion estimates

The additive CCD sensor noise is modeled as a white Gaussian process with zero mean and variance σ_η^2. Its effect on motion estimation is demonstrated in figures 2.14 and 2.15 with $\sigma_\eta = 0, 1, ..., 5$. For brevity we restrict ourselves to the 6 tap interpolation filter which offers the best accuracy for the estimated vectorfield. The performance for the nearest neighbour, bilinear and bicubic interpolation under noise can be found in the appendix.

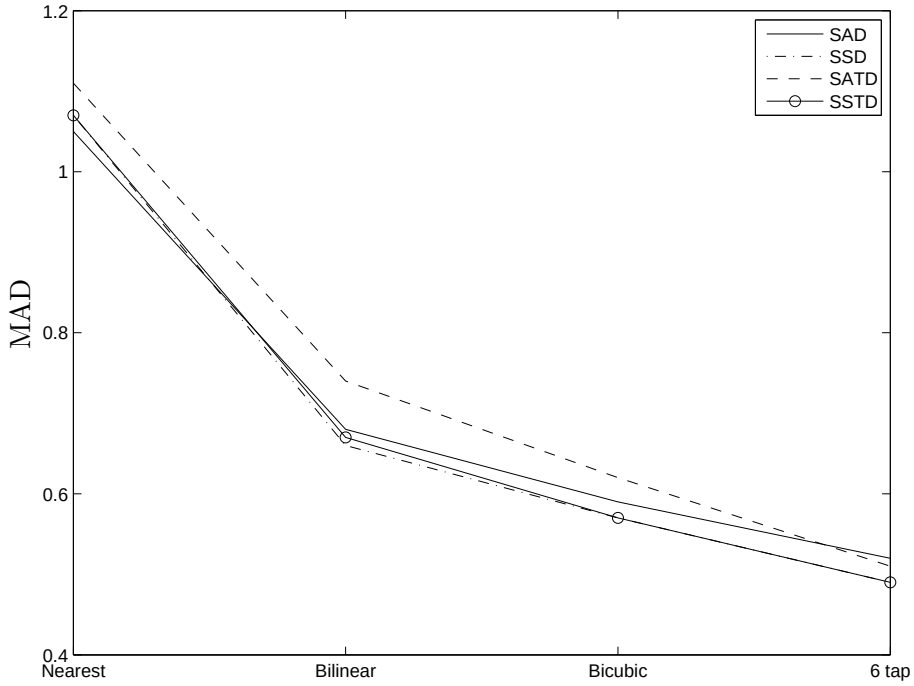


Figure 2.13: Simulation result: Effect of the interpolation scheme on motion estimation, 'Mobile' sequence

The destructive effect of noise on the motion estimation is clearly visible: Noise violates the assumption of intensity conservation, leading to an error in the estimated vectorfield. Furthermore it can again observed that the SSD and SSTD cost functions perform slightly better than the SAD and SATD cost function.

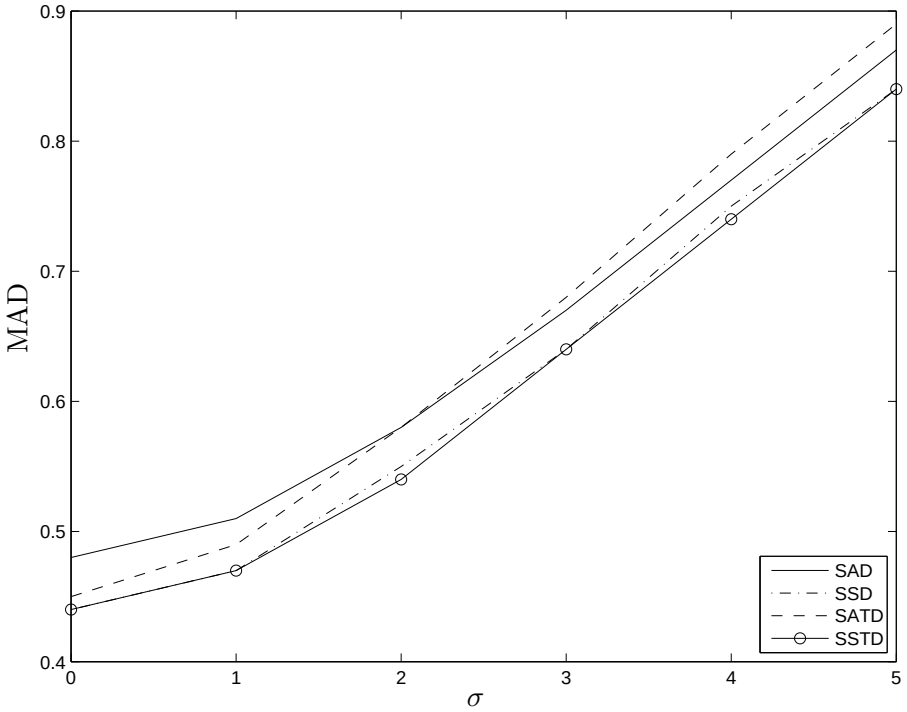


Figure 2.14: Simulation result: Effect of noise on motion estimation, 'Coastguard' sequence

2.4 Conclusion

In this chapter the classical block based motion estimation scheme for fractional pixel accuracy has been presented. Simulations have been run to show the effect of noise, aliasing, cost function and the interpolation scheme on the final estimate of the vectorfield. The following conclusions can be drawn:

- Aliasing has a highly destructive effect on motion estimation. Even advanced interpolation schemes can only in part compensate this.
- The occurrence of noise is a problem for motion estimation, especially for high-precision motion estimation with small blocksizes.
- The choice of the interpolation scheme is of importance for the motion estimate. Advanced schemes such as the bicubic interpolation or the 6 tap FIR interpolation filter scheme are preferred as they partly compensate the aliasing effect.
- The choice of the cost function has only a small influence on the quality of the estimated vectorfield. SSD and SSTD offer a slightly better performance when noise and aliasing are present compared to the SAD and SATD cost functions.

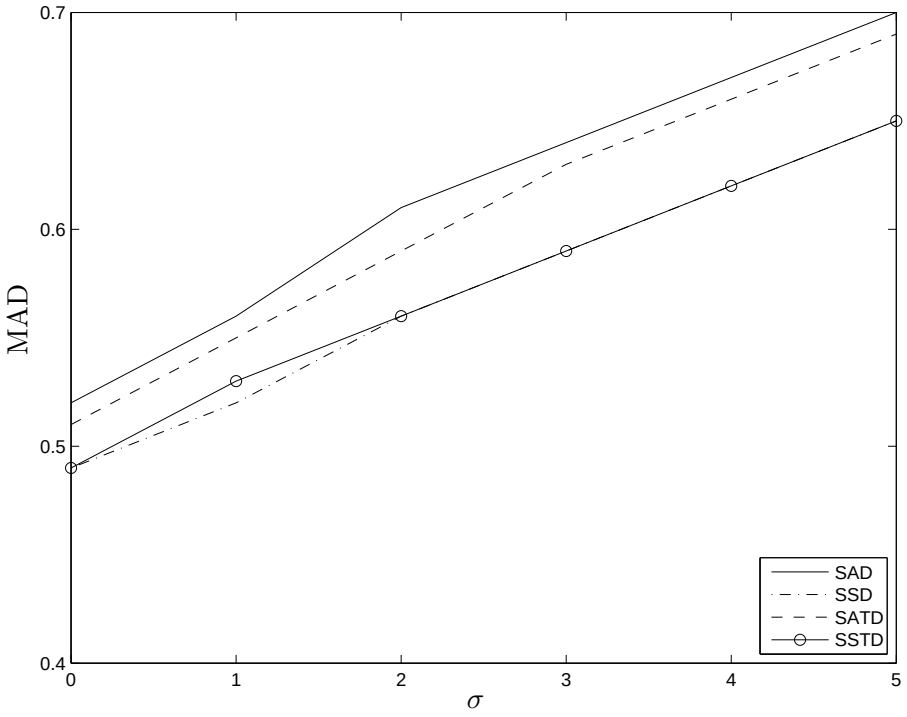


Figure 2.15: Simulation result: Effect of noise on motion estimation, 'Mobile' sequence

Chapter 3

Super-Resolution Image Reconstruction

In this chapter the concept of SR (Super-Resolution) is treated. Section 3.1 motivates the usage of SR techniques and shows some of their applications. In sections 3.2 - 3.7 the stochastic regularisation approach for Super-Resolution image reconstruction is presented in detail. Simulations for different SR methods are shown in section 3.8, whereas section 3.9 provides conclusions.

3.1 Super Resolution: Motivation

In many imaging applications there is a strong need for HR (High Resolution) images offering a more and more detailed view of the object of interest. Some classical applications are

- Medical imaging: Detailed images of high resolution are of importance in medicine, providing doctors a detailed view about the object of interest and thus helping to make a correct diagnosis.
- Satellite imaging: HR images are needed in satellite imaging applications to provide a detailed view of the target of interest.
- Surveillance: HR images in surveillance applications help e.g. to identify people or read licence plates of cars.

However the need for HR images is not only restricted to these latter professional applications. In consumer electronics, people want low cost digital cameras and camcorders providing higher and higher image resolution.
There are two straightforward approaches to increase the resolution of an image.

- Reducing the pixel size. Disadvantage: The amount of shot noise will increase. The physical limit to pixel size reduction has nearly been reached. [13]

- Increase the chip size. Disadvantage: Increases the capacitance which slows down the readout of the sensors.

Another disadvantage concerning these approaches is the increase in costs for high-precision optics and sensors. SR (Super Resolution) image reconstruction is an alternative approach for obtaining HR images and will be described in the following:
SR techniques describe methods to obtain a HR image out of a set of LR (Low Resolution) images. It is assumed that all LR images describe the same scene and are shifted, blurred, subsampled and noisy versions of a common underlying unknown HR image as depicted in figure 3.1. If the motion between the LR images can be described by fractional pixel displacements each LR image contains different information of the same HR image and thus SR reconstruction is possible. This concept of registration-based interpolation is shown in figure 3.2. In this figure three LR images (each of them has a resolution of 3×3) are shown. Given their relative displacement to each other a registration can be performed. Based on that registration a SR image with a higher resolution can be constructed by interpolation.

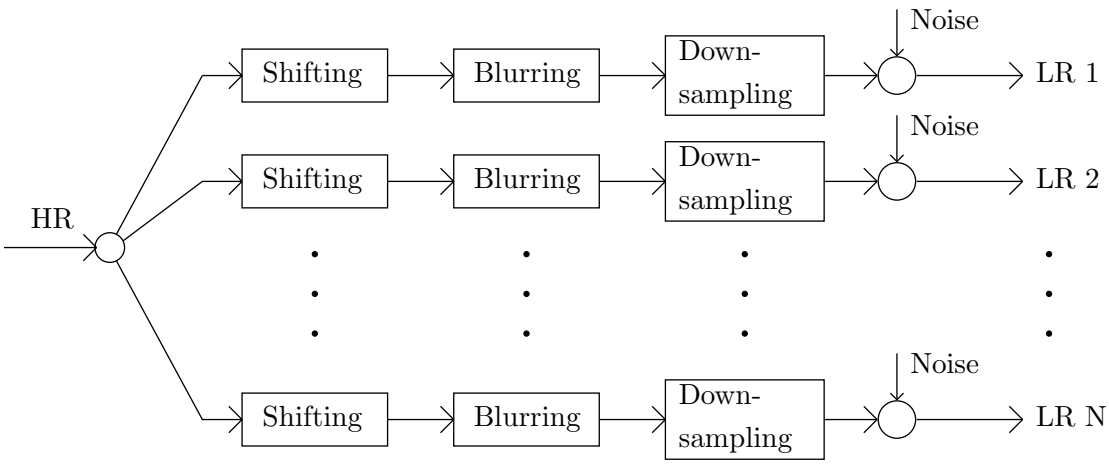


Figure 3.1: Modelling of the LR image capturing scheme

Figure 3.3 shows an example of three LR images with different subpixel displacements. Aliasing effects are visible in all parts of the image which contain high frequency parts, i.e. fine details such as the eyes of the pigs. It is the aim of SR to reconstruct a HR image out of several LR images which can be done by first applying motion estimation schemes (Registration), followed by an interpolation onto a finer grid and finally by performing a restoration in order to reduce blur and noise effects. This SR image retrieval scheme is visualised in figure 3.4

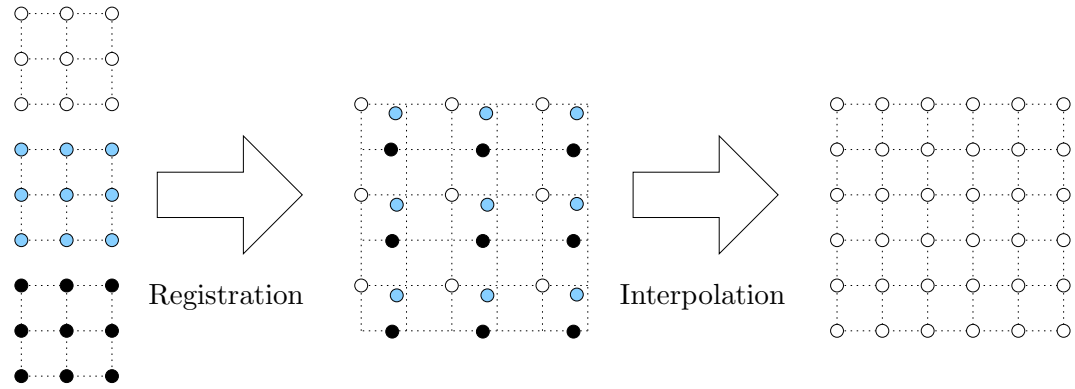


Figure 3.2: Registration and Interpolation

(a) LR_0 (b) LR_1 (c) LR_2

Figure 3.3: Three different LR versions of the same HR image

3.2 Stochastic regularisation approach

Assume the following observation model:

$$y_i = D \cdot B_i \cdot M_i \cdot x + \eta_i, \qquad i = 0, ..., F-1 \tag{3.1}$$

with

- y_i being the vectorised ith LR image denoted in lexicographic notation
- D being the downsampling matrix
- B_i being the blurring matrix for the ith image
- M_i being the motion matrix describing the spatial displacement between the images
- x being the underlying HR image in lexicographic notation
- η being the ith noise vector in lexicographic notation

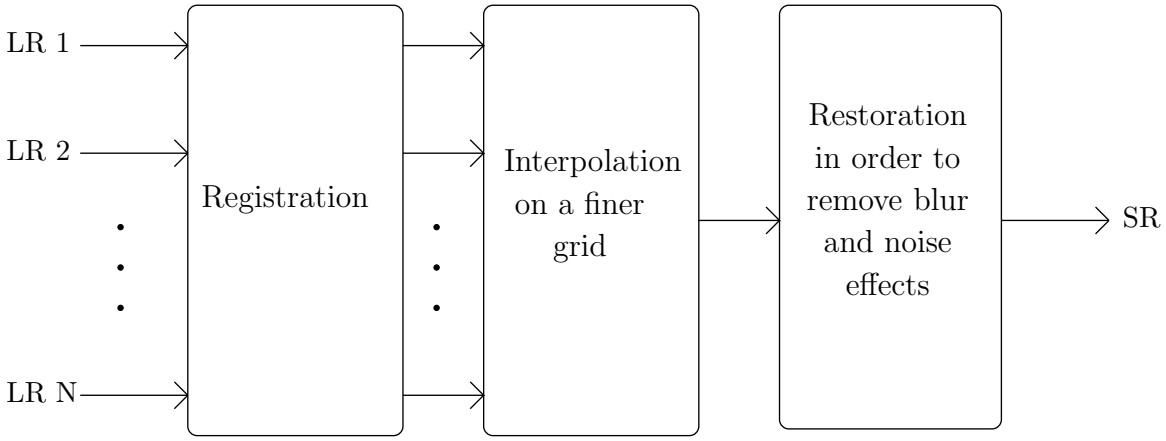


Figure 3.4: SR image retrieval scheme

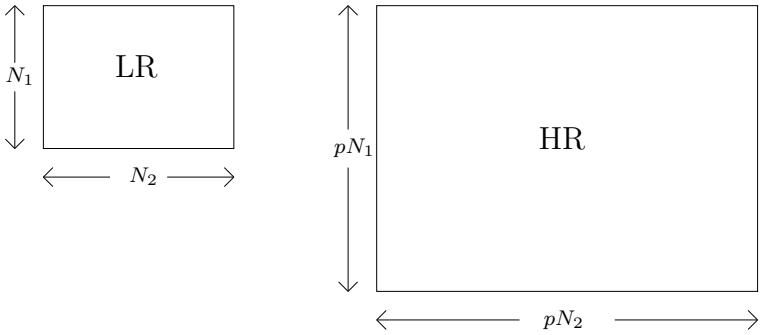


Figure 3.5: Image size for LR and HR images

Equation 3.1 describes mathematically what has been stated in the previous section: The outcome of video capturing is a set of LR images which are shifted, downsampled, blurred and noisy versions of the same underlying HR image. It is the aim of SR techniques to resolve the HR image out of the observed LR images, i.e. solve equation 3.1 for x. It is assumed that each LR image is of size $(N_2 \times N_1)$ and each HR image is magnified by a factor of p leading to a size of $(pN_2 \times pN_1)$ as visualised in figure 3.5.

Given equation 3.1 and the LR and HR image sizes as before table 3.1 shows the dimensions for the vectors and matrices:

$$\begin{array}{ccccc} [y_i] & = & (N_1 \cdot N_2) & \times & 1 \\ [D] & = & (N_1 \cdot N_2) & \times & (p^2 \cdot N_1 \cdot N_2) \\ [B_i] & = & (p^2 \cdot N_1 \cdot N_2) & \times & (p^2 \cdot N_1 \cdot N_2) \\ [M_i] & = & (p^2 \cdot N_1 \cdot N_2) & \times & (p^2 \cdot N_1 \cdot N_2) \\ [x] & = & (p^2 \cdot N_1 \cdot N_2) & \times & 1 \\ [\eta_i] & = & (N_1 \cdot N_2) & \times & 1 \end{array}$$

Table 3.1: Vector and matrix dimensions in the signal model

The signal model in vector/matrix form is visualised in equation 3.2.

$$\begin{pmatrix} \\ y_i \\ \\ \end{pmatrix} = \begin{pmatrix} \\ D \\ \\ \end{pmatrix} \begin{pmatrix} \\ B_i \\ \\ \end{pmatrix} \begin{pmatrix} \\ M_i \\ \\ \end{pmatrix} \begin{pmatrix} \\ x \\ \\ \end{pmatrix} + \begin{pmatrix} \\ \eta_i \\ \\ \end{pmatrix} \tag{3.2}$$

This concept of image-vectorisation and transforming with matrices is best shown with a small example ignoring the noise term in equation 3.1. For reasons of brevity an HR image size of (4x4) is chosen. x is then determined as the vectorisation of the HR image in lexicographic notation:

$$\text{HR Image} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}; \qquad x = \begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{41} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{42} \\ a_{13} \\ a_{23} \\ a_{33} \\ a_{43} \\ a_{14} \\ a_{24} \\ a_{34} \\ a_{44} \end{pmatrix}$$

The first transformation is due to the motion matrix M which in this example performs a global shift of $(0, 1)$, i.e. one pixel downwards. By shifting the image downwards by one pixel the lowest pixel row drops out and a new pixel row appears on the top. For simplicity it is assumed that the new pixel row is generated by pixel replication. This assumption is valid for large images but may be an inaccurate method for very small blocks. The resulting vector representing the shifted HR image is denoted

as x_S:

$$
\begin{aligned}
x_S &= M \cdot x \\
&= \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
\begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{41} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{42} \\ a_{13} \\ a_{23} \\ a_{33} \\ a_{43} \\ a_{14} \\ a_{24} \\ a_{34} \\ a_{44} \end{pmatrix}
= \begin{pmatrix} a_{11} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{12} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{13} \\ a_{13} \\ a_{23} \\ a_{33} \\ a_{14} \\ a_{14} \\ a_{24} \\ a_{34} \end{pmatrix}
\end{aligned}
$$

Now the shifted HR image is blurred by spatial averaging. Using a PSF (Point Spread Function) of $[0.5, 0.5]$ we arrive at the blurred and shifted HR image denoted as x_{SB}. Note that again, pixel replication has been used to generate missing pixel values beyond the image block borders.

$$
\begin{aligned}
x_{SB} &= B \cdot x_S \\
&= \frac{1}{4} \begin{pmatrix}
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4
\end{pmatrix}
\begin{pmatrix} a_{11} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{12} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{13} \\ a_{13} \\ a_{23} \\ a_{33} \\ a_{14} \\ a_{14} \\ a_{24} \\ a_{34} \end{pmatrix}
= \begin{pmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{41} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{42} \\ b_{13} \\ b_{23} \\ b_{33} \\ b_{43} \\ b_{14} \\ b_{24} \\ b_{34} \\ b_{44} \end{pmatrix}
\end{aligned}
$$

The blurred and shifted HR image x_{SB} is now downsampled using the matrix D leading to the LR image y:

$$
\begin{aligned}
y &= D \cdot x_{SB} \\
&= \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
b_{11} \\ b_{21} \\ b_{31} \\ b_{41} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{42} \\ b_{13} \\ b_{23} \\ b_{33} \\ b_{43} \\ b_{14} \\ b_{24} \\ b_{34} \\ b_{44}
\end{pmatrix}
= \begin{pmatrix} b_{11} \\ b_{22} \\ b_{33} \\ b_{44} \end{pmatrix}
\end{aligned}
$$

with

$$
\begin{aligned}
b_{11} &= \frac{1}{2} \cdot (a_{11} + a_{12}) \qquad b_{22} = \frac{1}{4} \cdot (a_{21} + a_{31} + a_{22} + a_{32}) \\
b_{33} &= \frac{1}{4} \cdot (a_{13} + a_{14}); \quad b_{44} = \frac{1}{4} \cdot (a_{23} + a_{33} + a_{24} + a_{34})
\end{aligned}
$$

The matrices in the signal model above may become very big (for a typical image size of 720×480 pixels and an enlargement factor of $p = 2$ the blurring and motion matrix have a size of $(1382400 \times 1382400) \approx 2 \cdot 10^{12}$ elements). However there are some properties which make the calculations feasible:

- All matrices D, B_i and M_i are sparse matrices - containing only very few non-zero elements
- D is a block circulant matrix containing only zeros and ones
- Assuming $B_i = B \ \forall \ i$ and only translational motion, B and M_i are block circulant matrices
- If a simple translational motion model is used the only elements in M_i are zeros and ones

3.3 Bayesian framework

One possible approach to solve the inverse Super-Resolution problem, i.e. solving equation 3.1 for x is the usage of a Bayesian framework. Let $P[x|y_0, ...y_{F-1}]$ be the conditional probability of the HR image x given the observed set of LR images $y_0, ...y_{F-1}$. This term has to be maximised in order to find the most probable HR image x. Using Bayes' law we obtain

$$P[x|y_0, ..., y_{F-1}] = \frac{P[y_0, ..., y_{F-1}|x] \cdot P[x]}{P[y_0, ...y_{F-1}]} \tag{3.3}$$

As the denominator in equation 3.3 does not affect the maximisation it is sufficient to maximise

$$\hat{x}_B = \arg\max_x \{P[y_0, ..., y_{F-1}|x] \cdot P[x]\} \tag{3.4}$$

3.4 Maximum Likelihood approach

The Maximum Likelihood (ML) approach is a special case in the Bayesian framework where $P[x]$ in equation 3.3 is assumed to be constant, i.e. all HR images are qually likely. Assuming the additive noise η_i in equation 3.1 to be independent and Gaussian distributed with zero mean and variance $\sigma^2_{\eta,i}$ leads to the probability density function

$$f_{\eta_i}(u) = \frac{1}{(2\pi\sigma^2_{\eta,i})^{\frac{N_1 N_2}{2}}} e^{-\frac{u^2}{2\sigma^2_{\eta,i}}} \tag{3.5}$$

The maximum likelihood estimator maximises $P[y_0, ..., y_{F-1}|x]$ as follows:

$$\begin{aligned} \hat{x}_{ML} &= \arg\max_x \{P[y_0, ..., y_{F-1}|x]\} \\ &= \arg\max_x \left\{ \prod_{i=0}^{F-1} P[y_i|x] \right\} \end{aligned} \tag{3.6}$$

Taking the negative log leads to

$$\begin{aligned} \hat{x}_{ML} &= \arg\min_x \left\{ -\ln \left(\prod_{i=0}^{F-1} P[y_i|x] \right) \right\} \\ &= \arg\min_x \left\{ \sum_{i=0}^{F-1} -\ln \left(P[y_i|x] \right) \right\} \end{aligned} \tag{3.7}$$

Given the signal model in equation 3.1 $P[y_i|x]$ - the probability of the LR image y_i given the HR image x - can be computed as:

$$P[y_i|x] = \frac{1}{(2\pi\sigma^2_{\eta,i})^{\frac{N_1 N_2}{2}}} e^{-\frac{(y_i - DB_i M_i x)^T (y_i - DB_i M_i x)}{2\sigma^2_\eta}} \tag{3.8}$$

Equation 3.8 states that the probability of a LR image y_i given the HR image x is highest when $y_i = DB_iM_ix$ (in this case y_i can be obtained out of x by applying the noisefree signal model) . $y_i - DB_iM_ix$ is a distance measure between the obtained LR image y_i and the simulated LR image DB_iM_ix. If these two terms are close (in the mean square sense as the noise η_i is assumed to be Gaussian distributed) the probability $P[y_i|x]$ is high, i.e. it is very likely that x is the true underlying HR image. On the other hand, if the distance between y_i and DBM_ix is high it is very unlikely that x is the desired HR image. Inserting equation 3.8 in equation 3.7 leads to

$$\begin{aligned}\hat{x}_{ML} &= \arg\min_x \left\{ \sum_{i=0}^{F-1} -\ln\left(\frac{1}{(2\pi\sigma_{\eta,i}^2)^{\frac{N_1N_2}{2}}} e^{-\frac{(y_i - DB_iM_ix)^T(y_i - DB_iM_ix)}{2\sigma_{\eta,i}^2}} \right) \right\} \\ &= \arg\min_x \left\{ \sum_{i=0}^{F-1} (y_i - DBM_ix)^T W_i^{-1} (y_i - DBM_ix) \right\} \end{aligned} \tag{3.9}$$

with W_i being the covariance matrix of the noise. For white noise we obtain $W_i = E\left\{\eta_i\eta_i^T\right\} = \sigma_{\eta,i}^2 I$. Taking the derivative of equation 3.9 with respect to x and setting it equal to zero leads to:

$$\sum_{i=0}^{F-1} \frac{1}{\sigma_{\eta,i}^2} (DB_iM_i)^T (y_i - DB_iM_ix) = 0 \tag{3.10}$$

Requiring the images $y_0, ..., y_{F-1}$ to be close in time, η_i can be assumed to be stationary within that time duration, leading to $\sigma_{n,i} = \sigma_n \ \forall \ i$. Equation 3.10 then reduces to:

$$\begin{aligned} \sum_{i=0}^{F-1} (DB_iM_i)^T (y_i - DB_iM_ix) &= 0 \\ \sum_{i=0}^{F-1} M_i^T B_i^T D^T (y_i - DB_iM_ix) &= 0 \\ \left(\sum_{i=0}^{F-1} M_i^T B_i^T D^T DB_iM_i \right) x &= \sum_{i=0}^{F-1} M_i^T B_i^T D^T y_i \end{aligned} \tag{3.11}$$

with

$$A_{ML} = \sum_{i=0}^{F-1} M_i^T B_i^T D^T DB_iM_i \qquad \text{and} \qquad b_{ML} = \sum_{i=0}^{F-1} M_i^T B_i^T D^T y_i \tag{3.12}$$

the optimisation problem can be written as

$$A_{ML} \cdot x = b_{ML} \tag{3.13}$$

Note that equation 3.13 is an underdetermined equation system which cannot be solved directly ($x = A_{ML}^{-1} b_{ML}$) as the determinant of A_{ML} equals zero. Optimisation algorithms such as the Steepest Descent or the Conjugate Gradient have to be applied here.

3.5 Maximum A Posteriori approach

In the ML approach it is implicitly assumed that all HR images are equally likely (i.e. $P[x]$ in equation 3.3 is constant). However in reality this assumption generally is not valid. The drawback of the ML approach will be shown with a small example, reducing the SR problem to a pure image reconstruction problem, i.e. obtaining a HR image out of one LR image. Figure 3.6 shows a 2×2 block for which a 4×4 HR version is desired. Using the ML approach each HR image which - blurred and down-sampled - is equal to the LR image fulfils equation 3.13 and thus is a HR candidate. As the optimisation problem in most practical cases is underdetermined there are a huge number of possible HR images x which all fulfil the requirement $y = DBx$. In figure 3.7 four possible HR versions of the block in figure 3.6 are shown which all are valid outcomes of the ML approach.

However, when comparing the HR versions in figure 3.7 with the LR image in figure

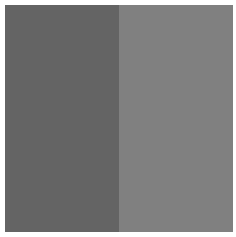


Figure 3.6: Example of a 2×2 LR block

(a) HR_1 (b) HR_2 (c) HR_3 (d) HR_4

Figure 3.7: Four possible HR blocks

3.6 the HVS would naturally choose HR_3 and HR_4 as being the most likely HR representatives, whereas HR_1 and HR_2 seem to be quite unlikely HR versions of the original LR block. This intuition is based on the fact that most natural images offer a characteristic which in the literature is known as 'global smoothness' [20]. This characteristic can be statistically described by Markov Random Fields [25].

Let N be a neighbouring matrix replacing each pixel value of the image with a representative value of its neighbourhood. Within this thesis only the 4-neighbourhood [25] will be considered with the mean being its representative value. In figure 3.8 an example of applying the neighbouring matrix has been shown. In this figure the matrix representations of x (first frame of the 'Mobile' sequence), Nx (an image in

which each pixel is replaced by the mean value of its 4-neighbourhood) and $x - Nx$ (an image in which from each pixel the mean value of its neighbourhood is subtracted) are shown. By means of random fields the 'global smoothness' requirement can be fulfilled by choosing an appropriate score function for $x - Nx$ which penalises large deviations between a pixel's value and the mean value of its neighbourhood.

3.5.1 MAP approach using a Gaussian prior

The GMRF (Gaussian Markov Random Field) [15] is due to its mathematical tractability an often used model for images and can be formulated as

$$(x - Nx) \sim \mathcal{N}(0, \sigma_G^2) \tag{3.14}$$

Equation 3.14 states that the difference between a pixel value and its neighbourhood is Gaussian distributed with zero mean and variance σ_G^2. This statistical property of the desired HR image can be incorporated in the Bayesian framework by a Gaussian prior:

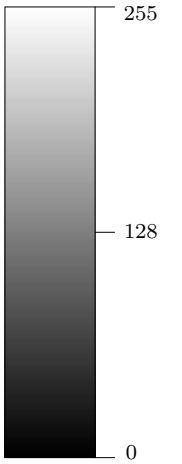
$$P_G[x] = \frac{1}{(2\pi\sigma_G^2)^{\frac{N_1 N_2}{2}}} \exp\left\{-\frac{(x - Nx)^T(x - Nx)}{2\sigma_G^2}\right\} \tag{3.15}$$

Making the same assumptions as for the ML approach (time- and space-invariant blurring, i.i.d. Gaussian noise) the MAP (Maximum A Posteriori) estimator with a Gaussian prior may then be written as

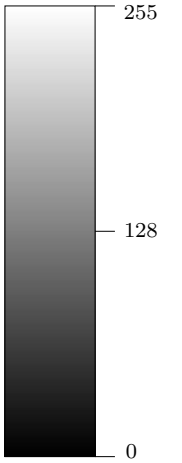
$$\begin{aligned}
\hat{x}_{MAP,G} &= \arg\max_x \left\{ \left(\prod_{i=0}^{F-1} P[y_i|x] \right) P_G[x] \right\} \\
&= \arg\min_x \left\{ -\ln \left(\prod_{i=0}^{F-1} P[y_i|x] \right) - \ln(P_G[x]) \right\} \\
&= \arg\min_x \left\{ \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + \frac{\sigma_\eta^2}{\sigma_G^2} (x - Nx)^T (x - Nx) \right\} \\
&= \arg\min_x \left\{ \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + c_G (x - Nx)^T (x - Nx) \right\}
\end{aligned}$$

with $c_G = \frac{\sigma_\eta^2}{\sigma_G^2}$ being the regularisation parameter. Taking the derivative of x and setting it equal to zero leads to

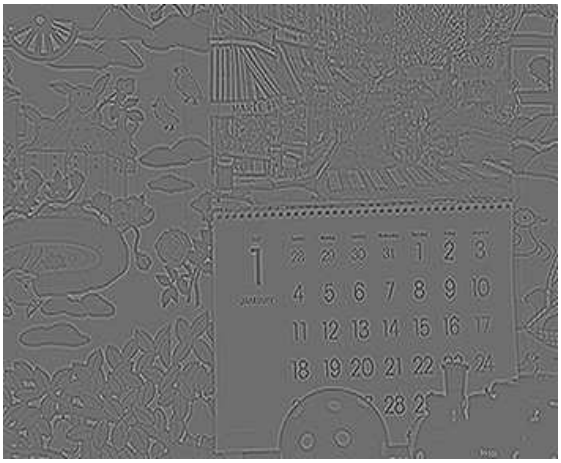
$$\begin{aligned}
-2 \sum_{i=0}^{F-1} (DBM_i)^T (y_i - DBM_i x) + 2c_G (I - N)^T (x - Nx) &= 0 \\
\left(\sum_{i=0}^{F-1} (DBM_i)^T DBM_i \right) x + c_G (I - N)^T (x - Nx) &= \sum_{i=0}^{F-1} (DBM_i)^T y_i
\end{aligned}$$

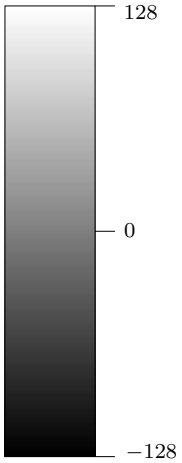


(a) x



(b) Nx





(c) $x - Nx$

Figure 3.8: Example: Neighbouring matrix

With

$$A_{MAP,G} = \sum_{i=0}^{F-1} (DBM_i)^T DBM_i + c_G (I-N)^T \cdot (I-N) \tag{3.16}$$

and

$$b_{MAP,G} = \sum_{i=0}^{F-1} (DBM_i)^T y_i \tag{3.17}$$

the optimisation problem can written in the same form as for the ML approach:

$$A_{MAP,G} \cdot x = b_{MAP,G} \tag{3.18}$$

The regularisation parameter c_G is dependent of the noise level σ_η^2 and the variance of the Gaussian prior, σ_G^2. If these values are available an optimal value for c_G can be calculated.
Note that the MAP approach with a Gaussian prior reduces to the ML approach described in the previous section if $c_G = 0$ which corresponds to the noise-free case ($\sigma_\eta^2 = 0$).

3.5.2 MAP approach using a double exponential prior

It has been shown [25], [18] that for most real-world images the GMRF is an unacceptable model. Large differences between the values of neighbouring pixels are caused naturally by edges and structures which both are important parts of an image. Using a Gaussian prior, these large differences - and thus important features of an image - are strongly penalised resulting in oversmoothed super-resolved images. In figure 3.9 and 3.10 the histogram of $x - Nx$ for the first frame of the 'Coastguard' and 'Mobile' sequence are shown. The QQ-plots are shown in figure 3.11 and 3.12. It can be seen that the assumption of $x - Nx$ to be Gaussian distributed is far from reality. A more heavy-tailed distribution such as the double exponential distribution would lead to a better image model. The heavy tails of the double exponential distribution penalise large differences between neighbouring pixels much less severely and thus provide an edge-preserving feature.

The assumption of $x - Nx$ to follow a double exponential distribution can be incorporated in the Bayesian scheme by the following prior:

$$P_D[x] = \frac{1}{(2\sigma_D^2)^{\frac{N_1 N_2}{2}}} e^{-\sqrt{\frac{2}{\sigma_D^2}}|x-Nx|} \tag{3.19}$$

Making the same assumptions as for the ML approach (time- and space-invariant blurring, i.i.d. Gaussian noise) the MAP (Maximum A Posteriori) estimator with a

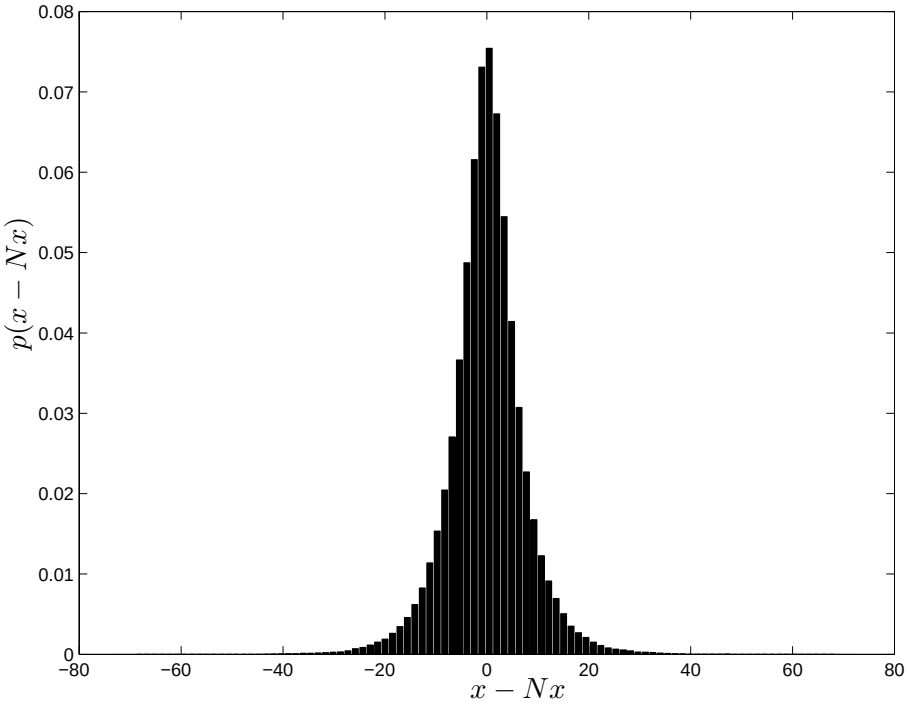


Figure 3.9: Example: Histogram of $x - Nx$, 'Coastguard' sequence

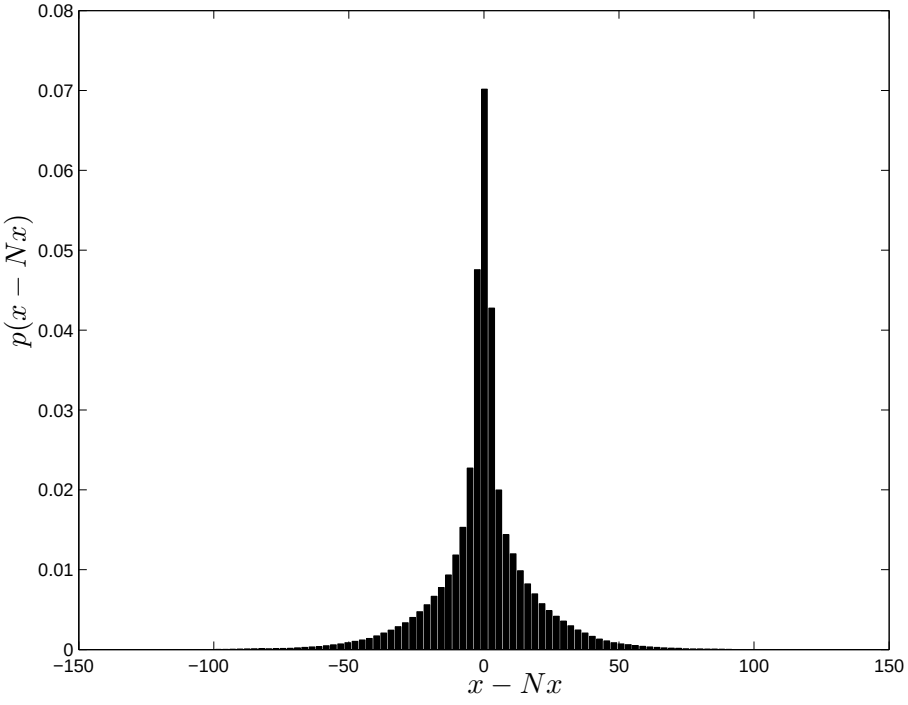


Figure 3.10: Example: Histogram of $x - Nx$, 'Mobile' sequence

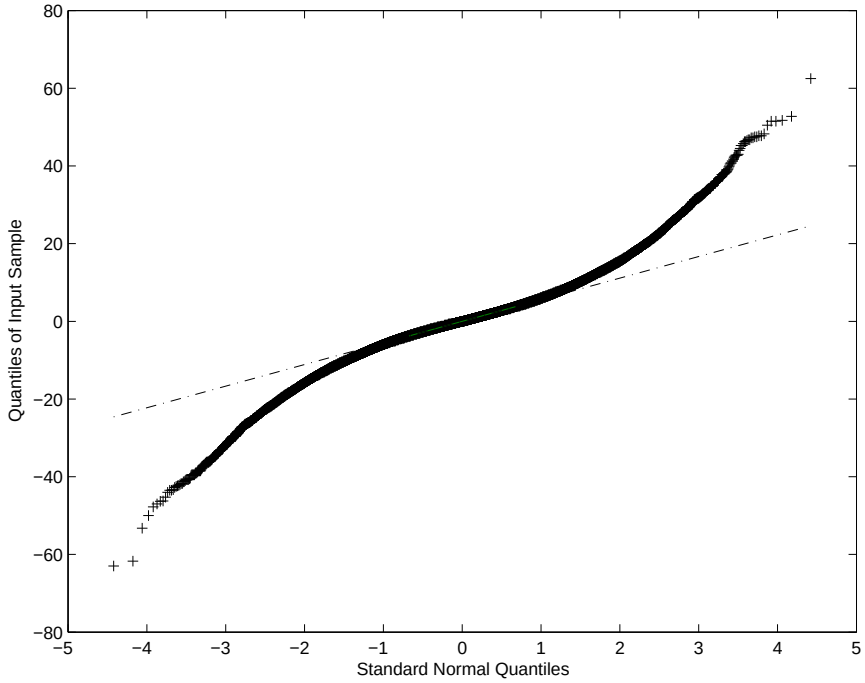


Figure 3.11: Example: QQ-plot of $x - Nx$, 'Coastguard' sequence

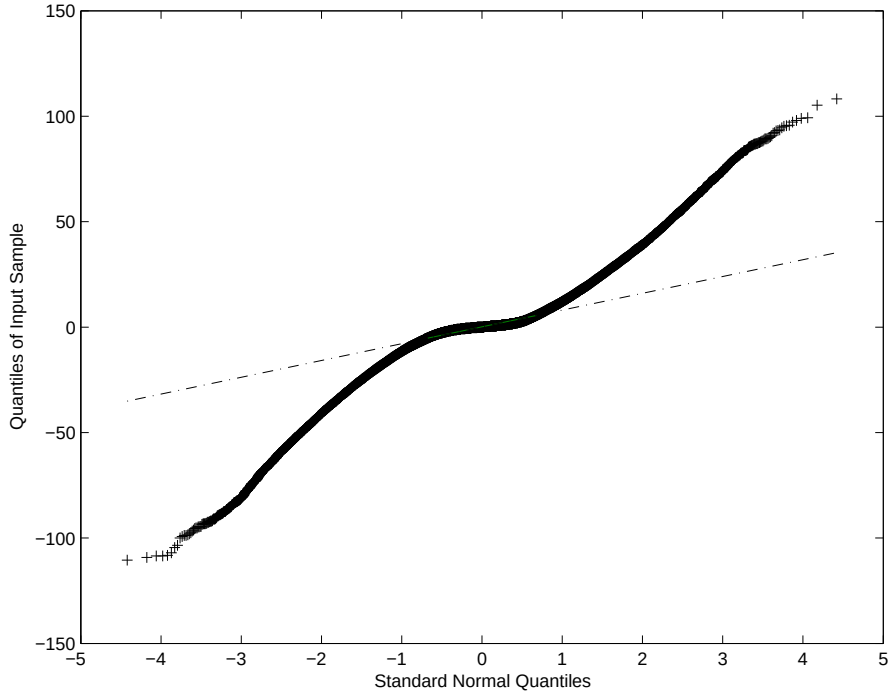


Figure 3.12: Example: QQ-plot of $x - Nx$, 'Mobile' sequence

double exponential prior may then be written as

$$\begin{aligned}
\hat{x}_{MAP,D} &= \arg\max_x \left\{ \left(\prod_{i=0}^{F-1} P[y_i|x] \right) P_D[x] \right\} \\
&= \arg\min_x \left\{ -\ln \left(\prod_{i=0}^{F-1} P[y_i|x] \right) - \ln(P_L[x]) \right\} \\
&= \arg\min_x \left\{ \frac{1}{\sigma_\eta^2} \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + \sqrt{\frac{2}{\sigma_D^2}} |x - Nx| \right\} \\
&= \arg\min_x \left\{ \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + \sqrt{2} \frac{\sigma_\eta^2}{\sigma_D} |x - Nx| \right\} \\
&= \arg\min_x \left\{ \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + c_D |x - Nx| \right\} \qquad (3.20)
\end{aligned}$$

with $c_D = \sqrt{2}\frac{\sigma_\eta^2}{\sigma_D}$ being the regularisation parameter for the double-exponential approach. Taking the derivative of x and setting it equal to zero leads to

$$-2 \sum_{i=0}^{F-1} (DBM_i)^T (y_i - DBM_i x) + c_D \frac{\partial}{\partial x} |x - Nx| = 0 \qquad (3.21)$$

In contrast to the MAP approach with a Gaussian prior and the ML approach equation 3.21 represents a nonlinear equation system which cannot be written in the form $Ax = b$. Therefore more complex optimisation algorithms have to be chosen to find a minimum of the cost function

$$F_{MAP,D} = \sum_{i=0}^{F-1} (y_i - DBM_i x)^T (y_i - DBM_i x) + c_D |x - Nx| \qquad (3.22)$$

3.6 Optimisation using the Conjugate Gradient algorithm

In this section the Conjugate Gradient (CG) algorithms will be discussed and applied to the problem of finding the most probable HR image x. The CG algorithm finds a local minimum by going in the negative direction of the gradient of the function to be minimised.

3.6.1 Linear optimisation

When using the ML approach or the MAP approach with a Gaussian prior the optimisation problem can be written in the form $Ax = b$, corresponding to a linear

equation system. Let $F(x)$ be the function to be minimised - which could be

$$F_{ML}(x) = \sum_{i=0}^{F-1}(y_i - DB_iM_ix)^T(y_i - DBM_ix) \tag{3.23}$$

for the maximum likelihood approach or

$$F_{MAP,G}(x) = \sum_{i=0}^{F-1}(y_i - DBM_ix)^T(y_i - DBM_ix) + c_G(x - Nx)^T(x - Nx) \tag{3.24}$$

for the maximum a posteriori approach with a Gaussian prior.
The CG algorithm starts with the following initialisation: Regarding the latter cost functions and their solutions in equation 3.13 and 3.18 the down gradient of $F(x)$ can be written as

$$g = \nabla F(x) = b - Ax \tag{3.25}$$

Given $\hat{x}^0$ being the approximated HR image in iteration step $r = 0$,e^0 is defined as being the gradient for the given $\hat{x}^0$, namely

$$e^0 = -g^0 = -\nabla F(\hat{x}^0) = A\hat{x}^0 - b \tag{3.26}$$

After having obtained the initialisation values for $\hat{x}^0$, e^0 and g^0 the next estimate $\hat{x}^{r+1}$ is calculated in the iteration step by going in the negative direction of e^r

$$\hat{x}^{r+1} = \hat{x}^r - \mu^r e^r \tag{3.27}$$

whereby μ^r is the stepsize in iteration step r determined as

$$\mu^r = \frac{(g^r)^T g^r}{(e^r)^T A e^r} \tag{3.28}$$

In each iteration step the residuum g^{r+1} and e^{r+1} are updated as follows

$$\begin{aligned} g^{r+1} &= g^r + \mu^r A e^r \\ e^{r+1} &= -g^{r+1} + \beta^r e^r \end{aligned} \tag{3.29}$$

with

$$\beta^r = \frac{(g^{r+1})^T g^{r+1}}{(g^r)^T g^r} \tag{3.30}$$

The iteration step is finished when the norm of the residuum, $|g^r|$ is below a predefined threshold ε, i.e. $|g^r| < \varepsilon$ with $\varepsilon \in \mathbb{R}^+$ or a maximum number of iterations R has been reached.
As an initial estimate $\hat{x}^0$, any vectorised image (e.g. a pure black image) may be taken. However an initial estimate close to the resulting HR image will speed up convergence which is why an interpolation of the first LR image y_0 (e.g. by applying the nearest neighbour interpolation or the bilinear interpolation) is preferred. In figure 3.13 the scheme for estimating a SR image out of two LR images using linear optimisation is shown. Given the vectorised LR images y_0 and y_1 and the motion matrix M_1 describing the motion between y_1 and y_0, the matrix A and vector b representing the linear equation system can be constructed. After that the CG algorithm can be applied for solving $Ax = b$ given an interpolated version of y_0 as the initial guess $\hat{x}^0$.

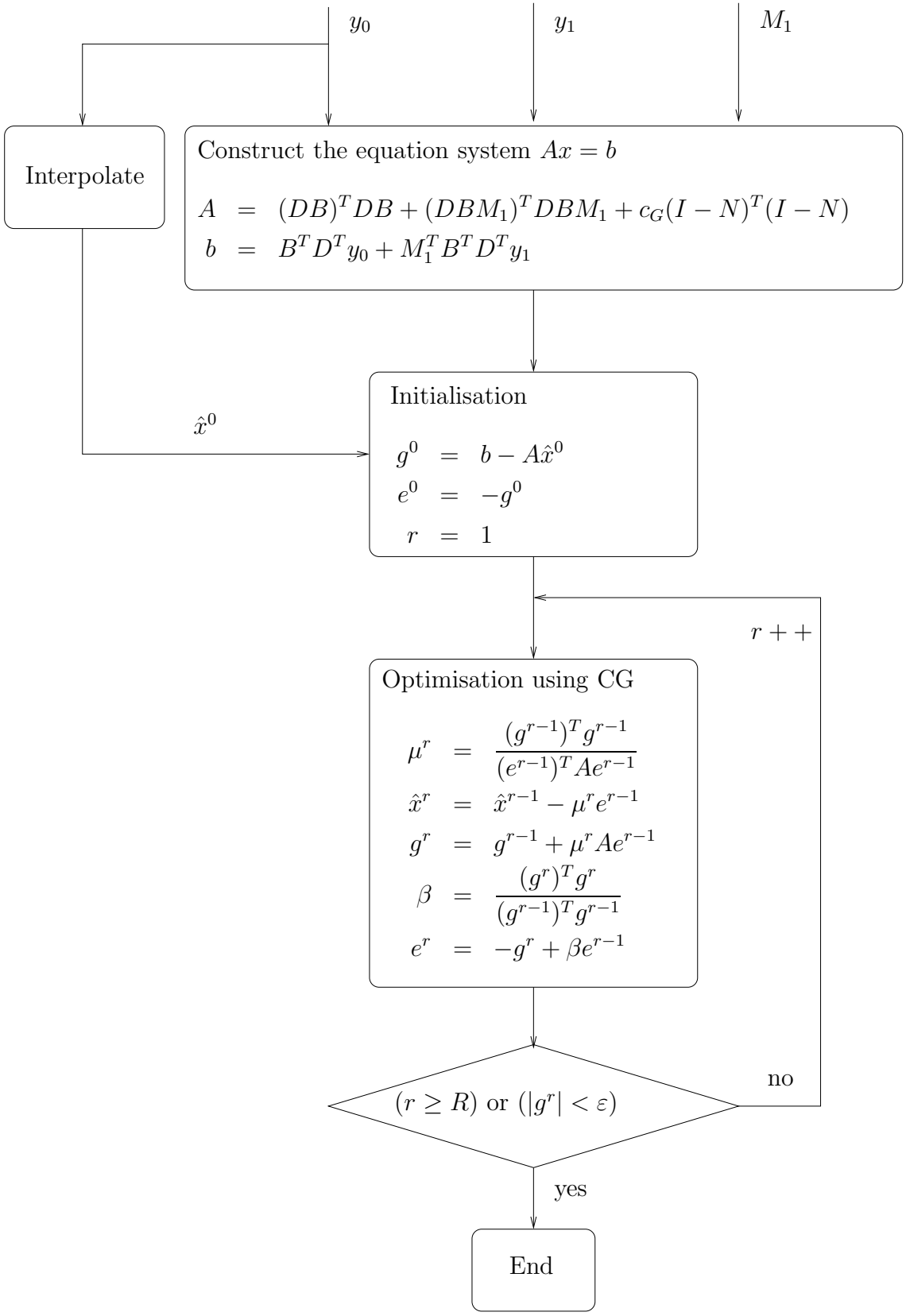


Figure 3.13: Block diagram for the SR estimation scheme using linear optimisation

3.6.2 Nonlinear optimisation

When using the MAP approach with a double exponential prior, nonlinear optimisation techniques have to be used to find the minimum of the cost function

$$F_{MAP,D} = \sum_{i=0}^{F-1}(y_i - DBM_i x)^T(y_i - DBM_i x) + c_D|x - Nx| \tag{3.31}$$

Using the CG algorithm for nonlinear functions requires some changes as the stepsize μ^r cannot be computed directly. Therefore the Newton-Raphson method is used to find a value for μ^r which minimises $F(x^r + \mu^r e^r)$. This corresponds to a line search of a non-quadratic function.
As a consequence, the nonlinear CG algorithm is more complex than the linear CG algorithm. Within each iteration step of the algorithm a line search has to be performed along the direction of the steepest descent which again requires a number of iterations and additional calculations.

3.7 Simulation setup and quality measures

In this section the simulation setup for Super-Resolution Image Reconstruction will be presented. Furthermore a set of measures to determine the quality of a SR image are discussed.
As explained in the previous sections the basic assumption of Super-Resolution Image Reconstruction states that subsequent images of a video sequence are shifted, blurred, downsampled and noisy versions of a common underlying HR image. In order to perform quality measures of different SR methods, a simulation scheme is needed to obtain LR images from a HR image. This scheme is shown in figure 3.14 for an artifical image. Starting with an HR image, shifting operations are applied, yielding a set of spatially displaced HR images. These shifted HR images are now blurred using a point spread function (PSF) of $[0.5, 0.5]$ and downsampled, yielding to a set of LR images with aliasing artifacts.

In the following some classical quality measures for images are treated. Let x be the original HR image in lexicographic notation with $x(n)$ being its n-th element, $n = 0, ..., p^2N_1N_2 - 1$. Let $\hat{x}$ be an estimate of x. Furthermore x and $\hat{x}$ are required to be z-bit images, i.e. $x(n) \in \mathbb{B}^z$ and $\hat{x}(n) \in \mathbb{B}^z$ with $B_{max} = 2^z - 1$ being the maximal pixel value. The following metrics for objective image and video quality have been established in the past:
Mean Absolute Error

$$\mathrm{MAE}(x, \hat{x}) = \frac{1}{p^2N_1N_2} \sum_{n=0}^{p^2N_1N_2-1} |x(n) - \hat{x}(n)| \tag{3.32}$$

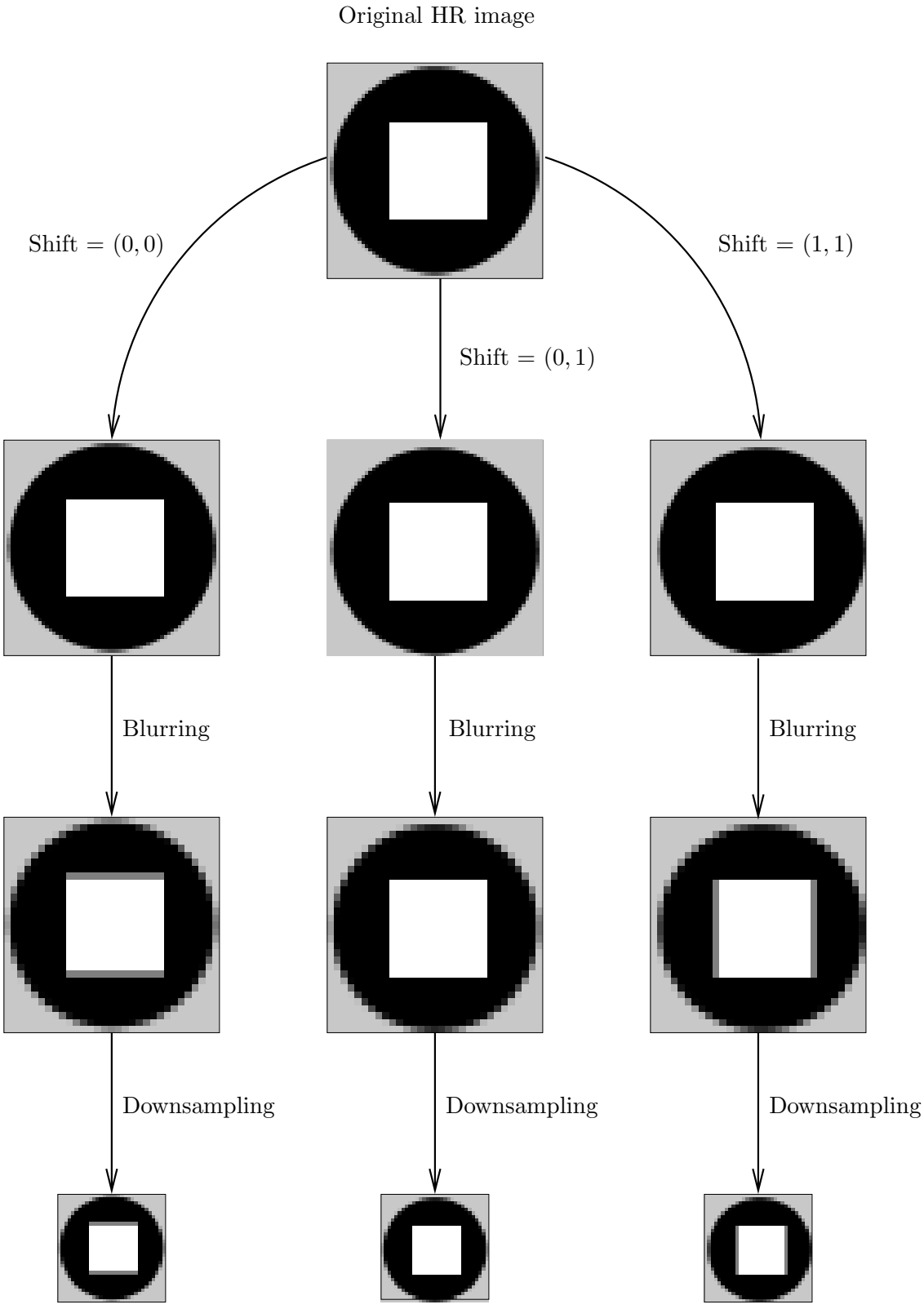


Figure 3.14: Simulation scheme for Super-Resolution Image Reconstruction

Mean Square Error

$$\mathrm{MSE}(x,\hat{x}) = \frac{1}{p^2 N_1 N_2} \sum_{n=0}^{p^2 N_1 N_2 - 1} (x(n) - \hat{x}(n))^2 \tag{3.33}$$

Peak Signal to Noise Ratio

$$\mathrm{PSNR}(x,\hat{x}) = 10 \cdot \log\left(\frac{B_{max}^2}{\mathrm{MSE}(x,\hat{x})}\right) \tag{3.34}$$

Beside the aforementioned basic image quality measures there has been a number of more sophisticated approaches within the last years providing a more meaningful index:

Universal Image Quality Index [23]

$$\mathrm{UIQI}(x,\hat{x}) = \frac{4\sigma_{x\hat{x}} \cdot \mu_x \mu_{\hat{x}}}{(\sigma_x^2 + \sigma_{\hat{x}}^2) \cdot (\mu_x^2 + \mu_{\hat{x}}^2)} \tag{3.35}$$

Structural Similarity Index [24]

$$\mathrm{SSIM}(x,\hat{x}) = \frac{(2\mu_x \mu_{\hat{x}} + C_1) \cdot (2\sigma_{x\hat{x}} + C_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + C_1) \cdot (\sigma_x^2 + \sigma_{\hat{x}}^2 + C_2)} \tag{3.36}$$

where

$$\mu_x = \frac{1}{N}\sum_{n=0}^{N-1} x(n), \qquad \mu_{\hat{x}}, = \frac{1}{N}\sum_{n=0}^{N-1} \hat{x}(n)$$

$$\sigma_x^2 = \frac{1}{N-1}\sum_{n=0}^{N-1} (x(n) - \mu_x)^2, \qquad \sigma_{\hat{x}}^2 = \frac{1}{N-1}\sum_{n=0}^{N-1} (\hat{x}(n) - \mu_{\hat{x}})^2$$

$$\sigma_{x\hat{x}} = \frac{1}{N-1}\sum_{n=0}^{N-1} (x(n) - \mu_x)(\hat{x}(n) - \mu_{\hat{x}})$$

$$C_1 = (K_1 \cdot B_{max})^2, \qquad C_2 = (K_2 \cdot B_{max})^2$$

with K_1 and K_2 being two constants (typically: $K_1 = 0.01$ and $K_2 = 0.03$).
Due to the non-stationarity of images and the space variance of image quality the latter two approaches are measured locally using a 8×8 sliding window (UIQI) or a 11×11 sliding window using a circular-symmetric Gaussian weighting function (SSIM). Beside the overall index (the mean of the UIQIs or SSIMs of all blocks covered by the sliding window) a quality map can be achieved showing the local quality index. UIQI and SSIM have been shown to provide a better representation of the error sensitivity of the HVS (Human Visual System) and thus a more meaningful measure of image quality compared with MSE or PSNR. Details can be found in [23] and [24].

3.8 Simulation results

In this section the effect of noise, the regularisation parameter c_G and c_D respectively, the initial estimate $\hat{x}^0$ and the number of iterations for the CG algorithm on the quality of the super-resolved image will be shown. We restrict ourselves to the PSNR and the SSIM as common measures for image quality. For the simulation setup as described in the previous section the luminance part of the first frames of the following YUV sequences will be used:

- **Coastguard**, 352×288 pixels
- **Foreman**, 352×288 pixels
- **Mobile**, 352×288 pixels
- **Paris**, 352×288 pixels

The following parameters will be used for simulating the downsampling process

- **Downsampling factor:** $p = 2$
- **Shift:** $(\frac{1}{2}, \frac{1}{2})$

For brevity in this section some representative plots and HR images are shown, the complete simulation results can be found in the appendix.

3.8.1 The effect of the initial estimate on the SR image

It has been stated in the previous section that an initial estimate which is close to the true HR image will speed up the convergence of the CG algorithm. In figure 3.15 the PSNR of the HR image with an increasing number of iterations is shown for the 'Mobile' sequence using the ML approach. The plot shows the PSNR increase when a pure gray image (pixel value 128 for all elements), a bilinear interpolated version of the first frame or an interpolated version of the first frame using the 6 tap interpolation filter presented in chapter 2 is used. It can be seen that an initial estimate using an advanced interpolation scheme should be preferred - e.g. to save iteration steps in the CG algorithm (which are very time-consuming) or to achieve a better quality given a fixed number of iterations. It can furthermore be seen that the CG algorithm ensures a good convergence. A saturation region is reached after $3 - 5$ iterations.

A similar result can be observed when using the SSIM as a measure of image quality (figure 3.16). Note that the SSIM index does not converge to its maximum value 1 as only two images are used for the SR image reconstruction. For a perfect reconstruction at least 4 images with different displacements are necessary, assuming

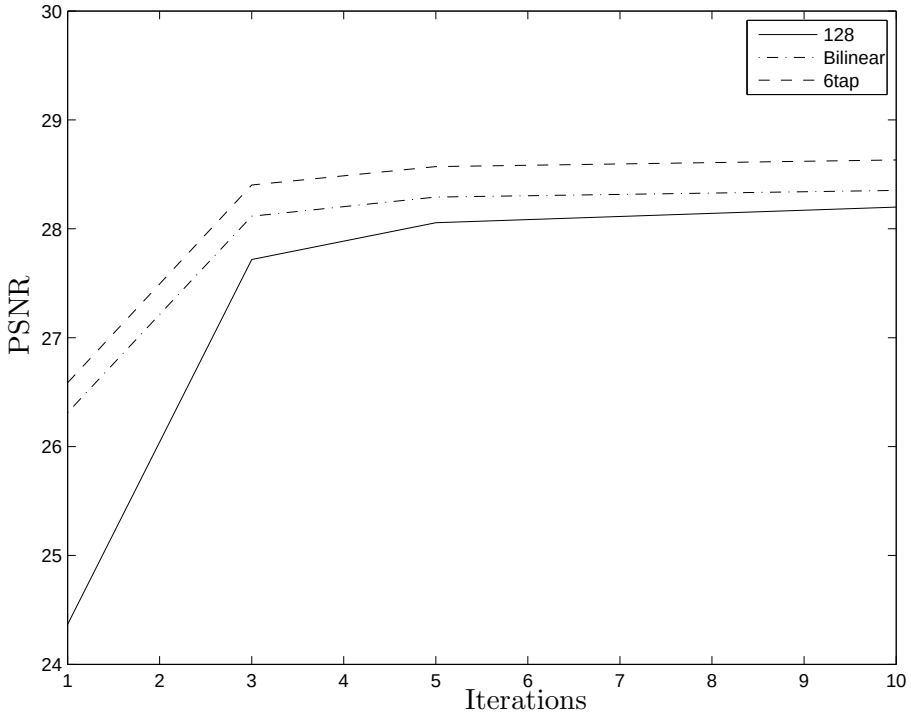


Figure 3.15: Simulation result: Effect of the initial estimate on Super-Resolution, 'Mobile' sequence, PSNR

a downsampling of $p = 2$.

In figure 3.17(a) a small part of the first frame of the 'Mobile' sequence is shown. In figure 3.17(b) and (c) the downsampled versions with a relative displacement of $(0.5, 0.5)$ are shown. These LR images are now used for image reconstruction. Figure 3.17(d) and (e) show the resulting HR images using the bilinear and 6 tap filter interpolation, figures 3.17(f)-(i) show the super-resolved HR image using $1, 3, 5$ and 10 iterations and the 6 tap filter interpolated image as the initial guess. An improvement in image quality can be observed when going from 1 to 3 iterations. However a further improvement when going to 5 or 10 iterations is hardly visible.

3.8.2 The effect of the regularisation on the SR image

For further examination the number of iterations is set to 5 (where a saturation region is reached as shown before) and the interpolated image using the 6 tap interpolation filter is used as the initial guess (which ensures fast convergence). The effect of the MAP prior (Gaussian or Double-Exponential) and the regularisation parameter will be examined in the following. In figures 3.18 and 3.19 the PSNR and SSIM of the super-resolved images using the 'Mobile' sequence are shown for different regularisation parameters, $0 \leq c \leq 1$. The following conclusions can be drawn:

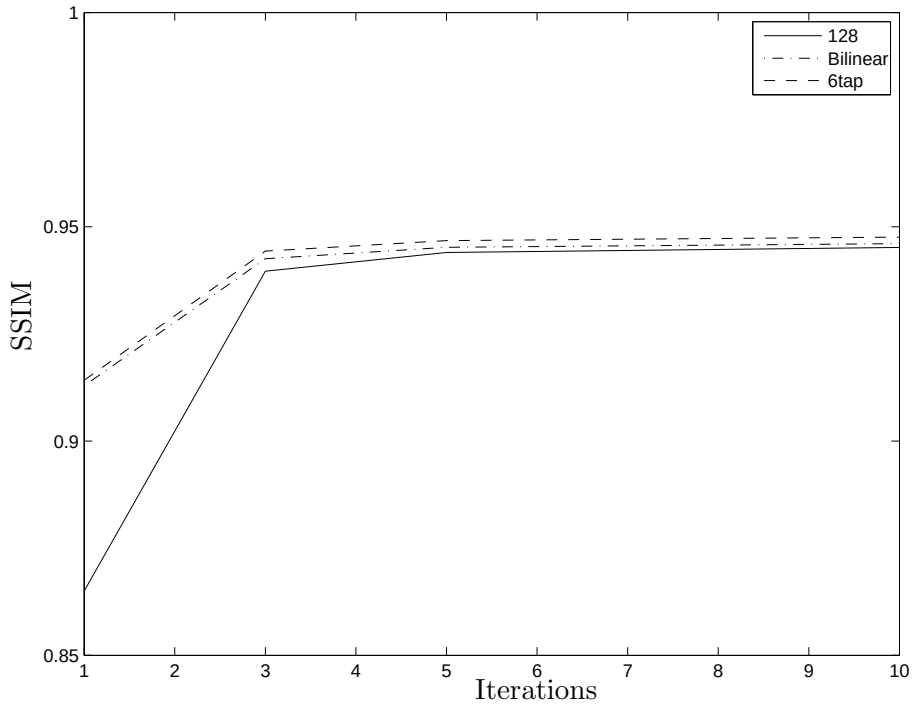
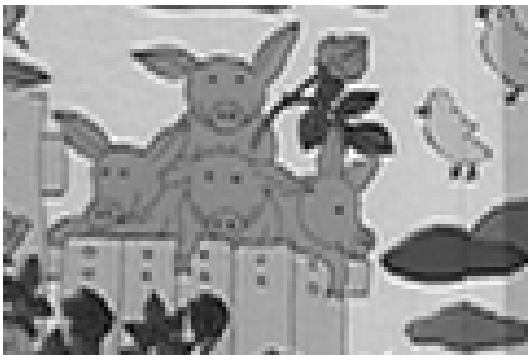


Figure 3.16: Simulation result: Effect of the initial estimate on Super-Resolution, 'Mobile' sequence, SSIM

- The objective quality (PSNR or SSIM) can be improved when regularisation is used

- The MAP with a Double-Exponential prior is more robust to changes of the regularisation parameter. When using the Gaussian prior (and thus an inaccurate image model) only small values for c should be chosen. When using the Double-Exponential prior (and thus a more accurate image model) the choice of c is less critical. This result is quite intuitive: The regularisation parameter controls the influence of the assumed statistical model for images. If the assumed model is inaccurate one should not rely on it and thus only a small value for c should be chosen. On the other side if the assumed image model is known to be accurate a higher value for c can be chosen.

In figure 3.20 a small part of the super-resolved images from the 'Mobile' sequence using a regularisation of $c = 0.5$ for the Gaussian and the Double-Exponential prior are shown. It can be seen that the Gaussian prior results in an oversmoothed image whereas the Double-Exponential prior provides an edge-preserving feature. This results in a far sharper image.



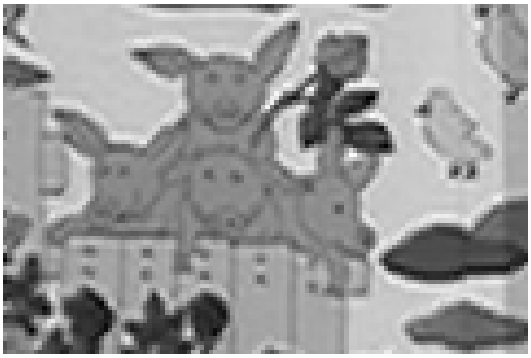
(a) Original image

(b) LR_1

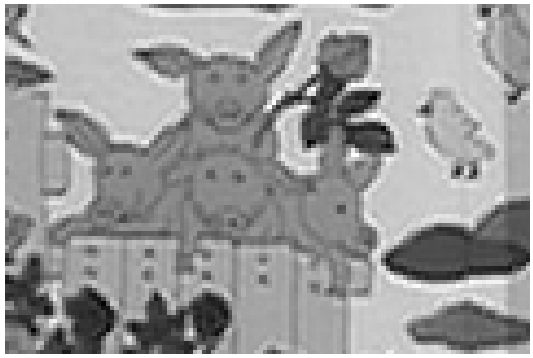
(c) LR_2

(d) Bilinear interpolation, PSNR = 20.66 dB

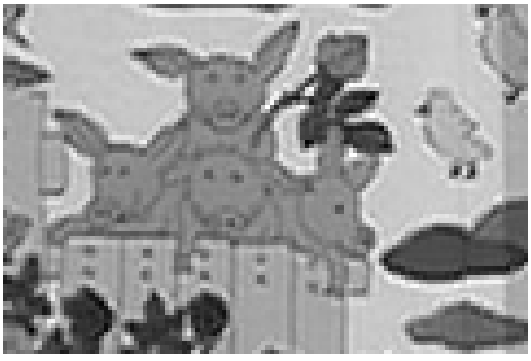
(e) 6tap filter interpolation, PSNR = 20.89 dB



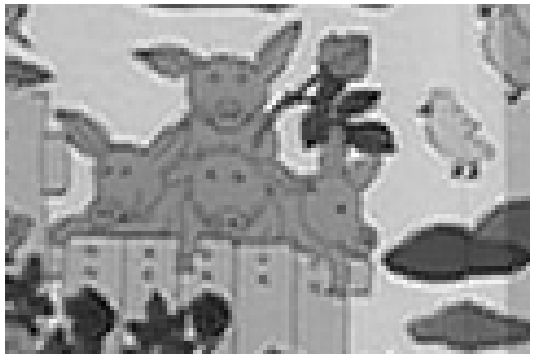
(f) SR, 1 Iteration, PSNR = 26.59 dB



(g) SR, 3 Iterations, PSNR = 28.4 dB



(h) SR, 5 Iteration, PSNR = 28.57 dB



(i) SR, 10 Iterations, PSNR = 28.63 dB

Figure 3.17: SR Example: Number of iterations vs. image quality

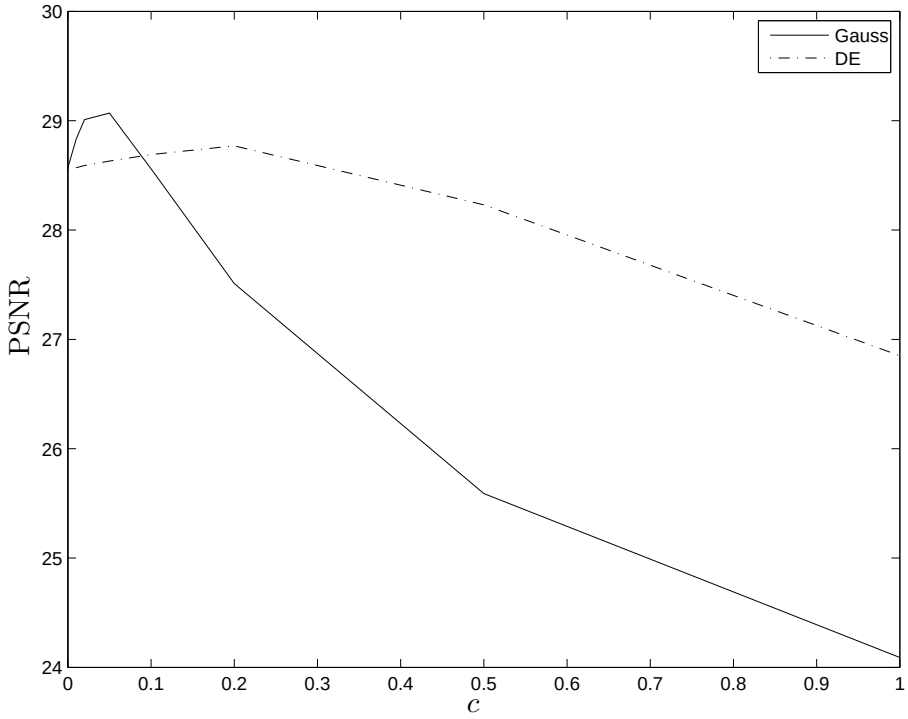


Figure 3.18: Simulation results: Effect of the regularisation on Super-Resolution, 'Mobile' sequence, PSNR

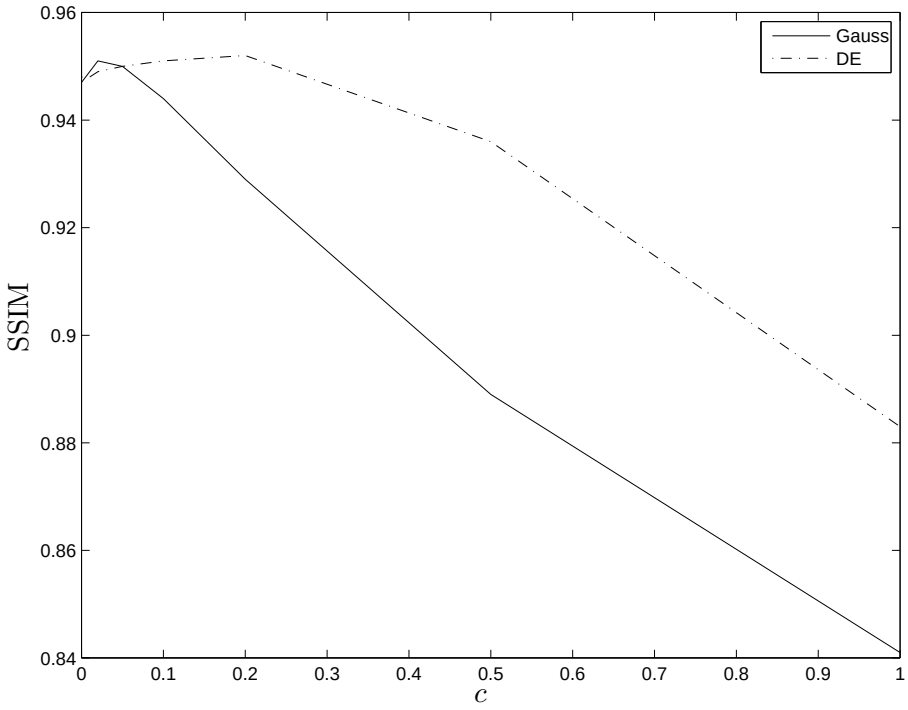
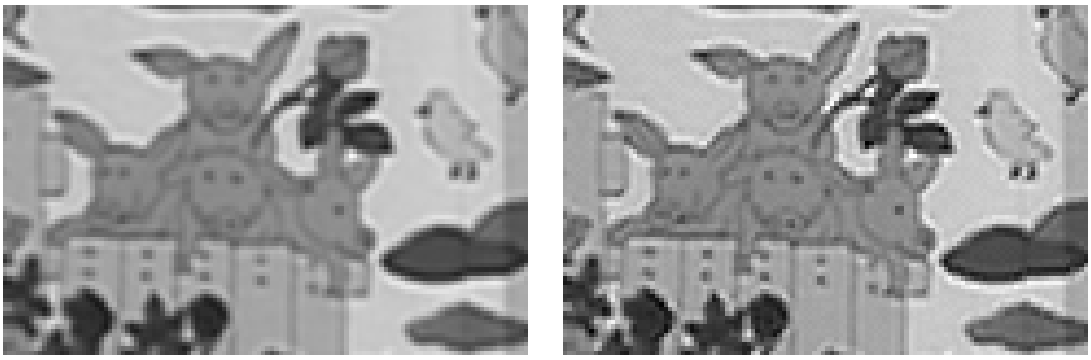


Figure 3.19: Simulation results: Effect of the regularisation on Super-Resolution, 'Mobile' sequence, SSIM



(a) Gaussian prior, PSNR = 25.59 dB

(b) Double-Exponential prior, PSNR = 28.23 dB

Figure 3.20: SR Example: Regularisation vs. image quality

3.8.3 The effect of noise on the SR image

The regularisation parameter is dependent on σ_η. It is therefore expected that a high value for the regularisation should be chosen when σ_η is large and regularisation should be kept at a small level if σ_η is small. Starting with the GMRF model, in table 3.2 the SSIM of the super-resolved images is shown for $\sigma_\eta = 0, ..., 5$ and c varying from 0 to 1. Simulation results have been averaged over 100 Monte Carlo simulations. For each value of σ_η the row which returns the highest SSIM is marked in bold letters. It can be seen that e.g. for the noise-free case a small regularisation parameter $c = 0.03$ is preferred whereas for $\sigma_\eta = 5$ a larger regularisation parameter ($c = 0.2$) should be preferred. When considering the double-exponential prior the SSIM of the super-resolved image can be seen in table 3.3. Again the positions which for a given σ_η return the highest SSIM are shown in bold letters. It can be seen that a far higher regularisation parameter can be chosen, starting from 0.1 for the noise-free case up to 0.7 for $\sigma_\eta = 5$. In practice it is very difficult to calculate the best value for c as one needs information of the noise level and statistics of the HR image. Both are generally unknown and difficult to estimate. It is therefore desirable that the quality of the super-resolved image is robust to deviations from the 'best' regularisation parameter.

	σ_η					
c	0	1	2	3	4	5
0	0.94679	0.93937	0.91817	0.88788	0.85365	0.81902
0.01	0.94913	0.94225	0.92249	0.89375	0.86128	0.82741
0.02	0.95063	0.94415	0.9252	0.89754	0.86628	0.83352
0.03	**0.95119**	0.945	0.92701	0.90067	0.87001	0.83851
0.04	0.95107	**0.94512**	0.92805	0.90285	0.87344	0.84268
0.05	0.95045	0.94479	0.9286	0.90457	0.87648	0.84648
0.06	0.94951	0.94411	**0.9288**	0.90588	0.87863	0.84957
0.07	0.94832	0.94319	0.92867	0.90671	0.8806	0.85226
0.08	0.94704	0.94209	0.92825	0.90719	0.88206	0.85479
0.09	0.94568	0.94097	0.9277	**0.90751**	0.88287	0.85651
0.1	0.94429	0.93969	0.92692	0.90748	**0.88385**	0.85774
0.2	0.92935	0.92586	0.91632	0.90148	0.88298	**0.86185**
0.3	0.91429	0.91142	0.90365	0.89152	0.87592	0.85794
0.4	0.90095	0.89851	0.89175	0.88118	0.86743	0.85153
0.5	0.88924	0.88701	0.88096	0.87141	0.859	0.84474
0.6	0.93731	0.93095	0.92006	0.90495	0.88399	0.85735
0.7	0.92413	0.92148	0.91843	0.90295	0.88241	0.8589
0.8	0.90908	0.90683	0.90271	0.89433	0.88097	0.85879
0.9	0.89444	0.89247	0.88873	0.8814	0.87072	0.85446
1	0.88319	0.88131	0.87671	0.8696	0.85947	0.8461

Table 3.2: Simulation results: Effect of noise on the SR image, 'Mobile' sequence, Gaussian prior, SSIM

	σ_η					
c	0	1	2	3	4	5
0	0.94767	0.93932	0.91832	0.88848	0.85481	0.81978
0.01	0.94813	0.93995	0.91908	0.88953	0.85533	0.82067
0.02	0.94854	0.94059	0.92	0.89038	0.85647	0.82179
0.03	0.94891	0.94116	0.9207	0.89143	0.85809	0.82295
0.04	0.94924	0.94165	0.92148	0.89239	0.85857	0.82339
0.05	0.94955	0.94213	0.92212	0.89288	0.85972	0.82436
0.06	0.94983	0.94256	0.92282	0.89396	0.86012	0.82548
0.07	0.9501	0.94296	0.92342	0.89474	0.86117	0.82676
0.08	0.95033	0.94333	0.92404	0.89559	0.86213	0.82743
0.09	0.95055	0.94367	0.92459	0.89615	0.86285	0.82794
0.1	0.95074	0.94398	0.92501	0.89684	0.86384	0.829
0.2	**0.95165**	0.946	0.9291	0.90247	0.87025	0.83564
0.3	0.95118	**0.94612**	**0.93138**	0.90646	0.87519	0.84128
0.4	0.94667	0.94281	0.93099	**0.90958**	0.88007	0.84669
0.5	0.93601	0.93291	0.92425	0.90815	**0.88427**	0.8543
0.6	0.93731	0.93095	0.92006	0.90495	0.88399	0.85735
0.7	0.92413	0.92148	0.91843	0.90295	0.88241	**0.8589**
0.8	0.90908	0.90683	0.90271	0.89433	0.88097	0.85879
0.9	0.89444	0.89247	0.88873	0.8814	0.87072	0.85446
1	0.88319	0.88131	0.87671	0.8696	0.85947	0.8461

Table 3.3: Simulation results: Effect of noise on the SR image, 'Mobile' sequence, DE prior, SSIM

3.9 Conclusions

In this chapter the concept of Super-Resolution Image Reconstruction has been treated. The following conclusions can be drawn:

- Motion estimation is the first and most important part for a successful SR image reconstruction requiring high-precision motion estimators.
- The Maximum-Likelihood approach is fast but may yield incorrect solutions far away from the original HR image due the ill-posed nature of the optimisation problem.
- Regularisation is crucial for SR image reconstruction, stabilising the optimisation problem and tending to solutions which are close to what the HVS would expect.
- Using the Gaussian prior yields to the optimisation of a linear equation system which can effectively be solved by the (linear) Conjugate Gradient algorithm. However as the GMRF is an inaccurate model the regularisation parameter should be kept small, otherwise this will result in oversmoothed solutions for the HR image.
- The Double-Exponential prior provides a better fit to the statistics of many images. It is therefore less susceptible to the choice of the regularisation parameter. Discontinuities are not as strongly penalised as for the Gaussian prior which results in sharper super-resolved images providing an edge-preserving feature. However when using the Double-Exponential prior a non-linear equation system has to be solved. This requires the computational costly use of non-linear optimisation algorithms.
- The regularisation parameter c is directly dependent on the noise level σ_η^2 and statistics of the HR image. Both quantities are difficult to estimate in practice. When using the Double-Exponential prior and thus a better image model, the quality of the super-resolved image is more robust with respect to deviations from an ideal regularisation parameter c than when using a Gaussian prior.

Chapter 4

Super-Resolution-based Motion Estimation

In this chapter the use of Super-Resolution techniques for motion estimation is treated. Section 4.1 gives a motivation for this approach. Section 4.2 provides a detailed description of the SR-based motion estimator, including its practical implementation. Simulation results comparing the new motion estimator with existing ones are shown in section 4.3 whereas section 4.4 provides conclusions.

4.1 Super-Resolution-based motion estimation: A motivation

Motion estimation and Super-Resolution image reconstruction in the past have mostly been treated as being two independent problems. However there is an interdependence between the problem of estimating motion information between one image and a reference image and the problem of reconstructing a SR image. Interpolation to a finer grid can only be done if there are accurate estimates of the motion. On the other hand, one can only get accurate (subpixel) motion estimates if there exist an aliasing- and noise-reduced reference image.
Motion estimators using the blockmatching technique as described in chapter 2 minimise cost functions such as the SAD (sum of absolute difference) or the SATD (sum of absolute transform differences), in order to find a corresponding block in the reference image which is closest (in the sense of the cost function) to the current block. Using these techniques one assumes two blocks to show the same part of a scene if their pixel intensity values are close (SAD) or if their transform coefficients (e.g. by using the DCT-transform) are close. However as shown in chapter 2 this measure may not be meaningful in the presence of aliasing and noise.
In order to find the true underlying motion a new approach will be presented here

stating that two blocks show the same part of a scene if they originate from the same Super-Resolution block.

4.2 The Super-Resolution based motion estimator

As in chapter 2 an image of interest will be divided into non-overlapping blocks $b_{1,0}, ..., b_{K-1}$ of equal quadratic size $Q \times Q$ as depicted in figure 4.1.

A block $b_{1,k}$ for which a motion vector is wanted is now compared to a set of

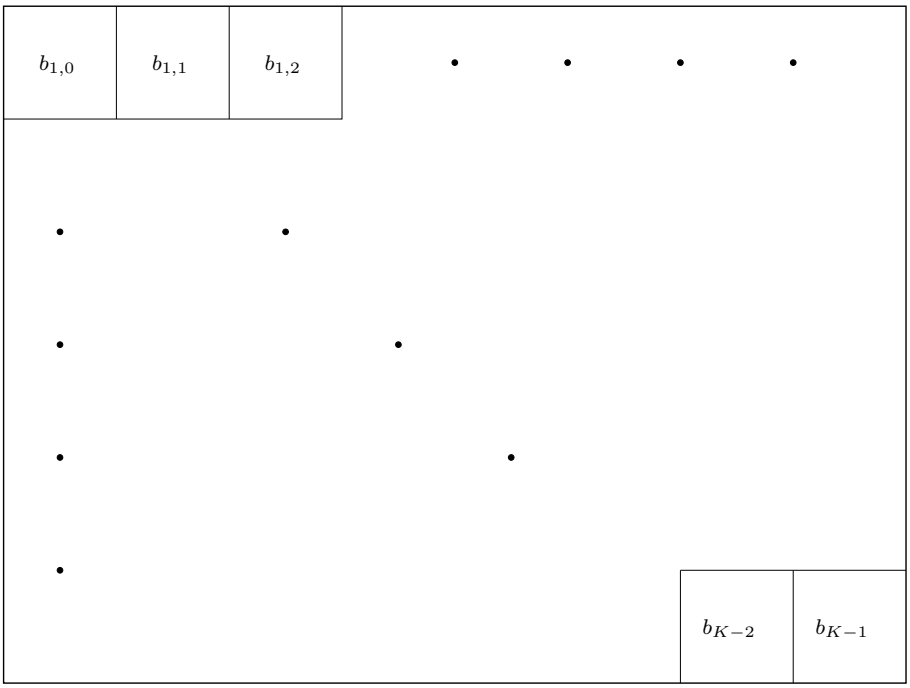


Figure 4.1: Image division of into blocks

overlapping blocks $b_{0,l}$ with $l = 0, ..., L-1$ in a specified region of the reference image as shown in figure 4.2.

Instead of performing an interpolation on the images as explained in chapter 2 and searching for the best matching block in the sense of a cost function such as SAD or SATD the Super-Resolution-based motion estimator performs a SR block reconstruction using for instance the ML approach, minimising

$$F_{ML}(x) = (b_{1,k} - DBx)^T(b_{1,k} - DBx) + (b_{0,l} - DBM^j x)^T(b_{0,l} - DBM^j x) \quad (4.1)$$

with x being the true underlying (and unknown) SR block. Note that in the first term the motion matrix is an identity matrix (as the motion from $b_{1,k}$ to itself is (0,0)) and can be dropped. Depending on the subpixel resolution to be achieved, one now chooses appropriate matrices M^j, $j = 1, ..., J$. It is assumed that $F_{ML}(x)$ will converge fast to a HR block x resulting in low costs when for the true underlying

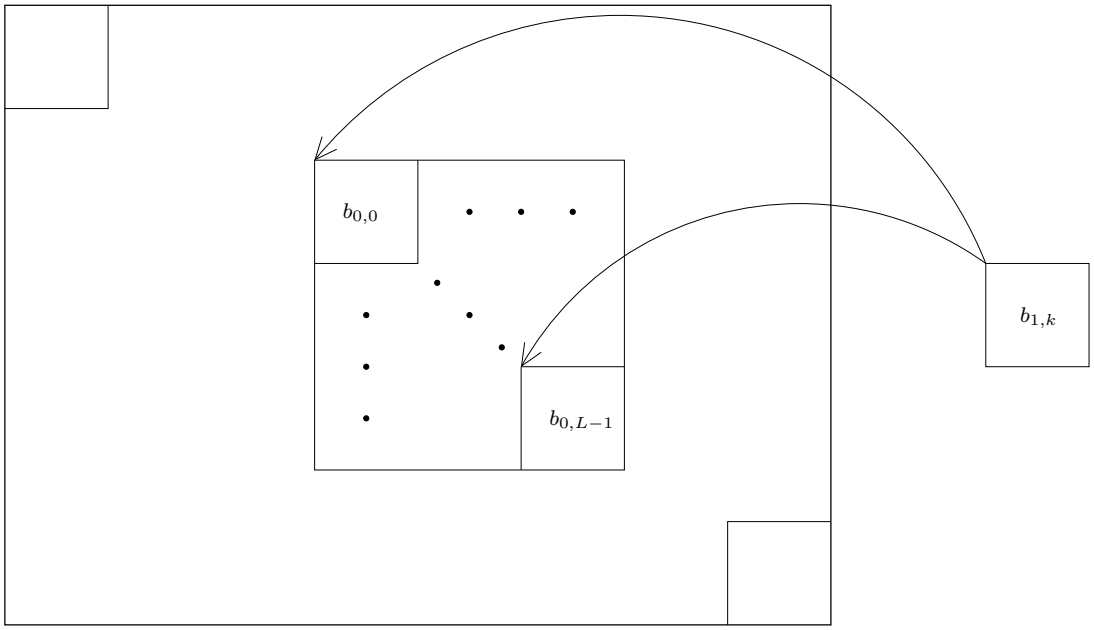
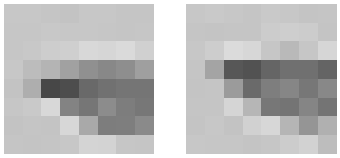


Figure 4.2: The blockmatching principle

motion matrix $M^{\tilde{j}}$.
This concept is best shown with a small example, considering only the maximum likelihood approach for the time being. In figure 4.3 two 8×8 blocks $b_{1,k}$ (the block for which a motion vector is searched) and $b_{0,l}$ (the reference block) are shown. The true motion between $b_{1,k}$ and $b_{0,l}$ is $(0, -\frac{1}{2})$, i.e. half a pixel upwards.
Restricting the motion vector resolution to 0.5 pixels there are in total nine matrices



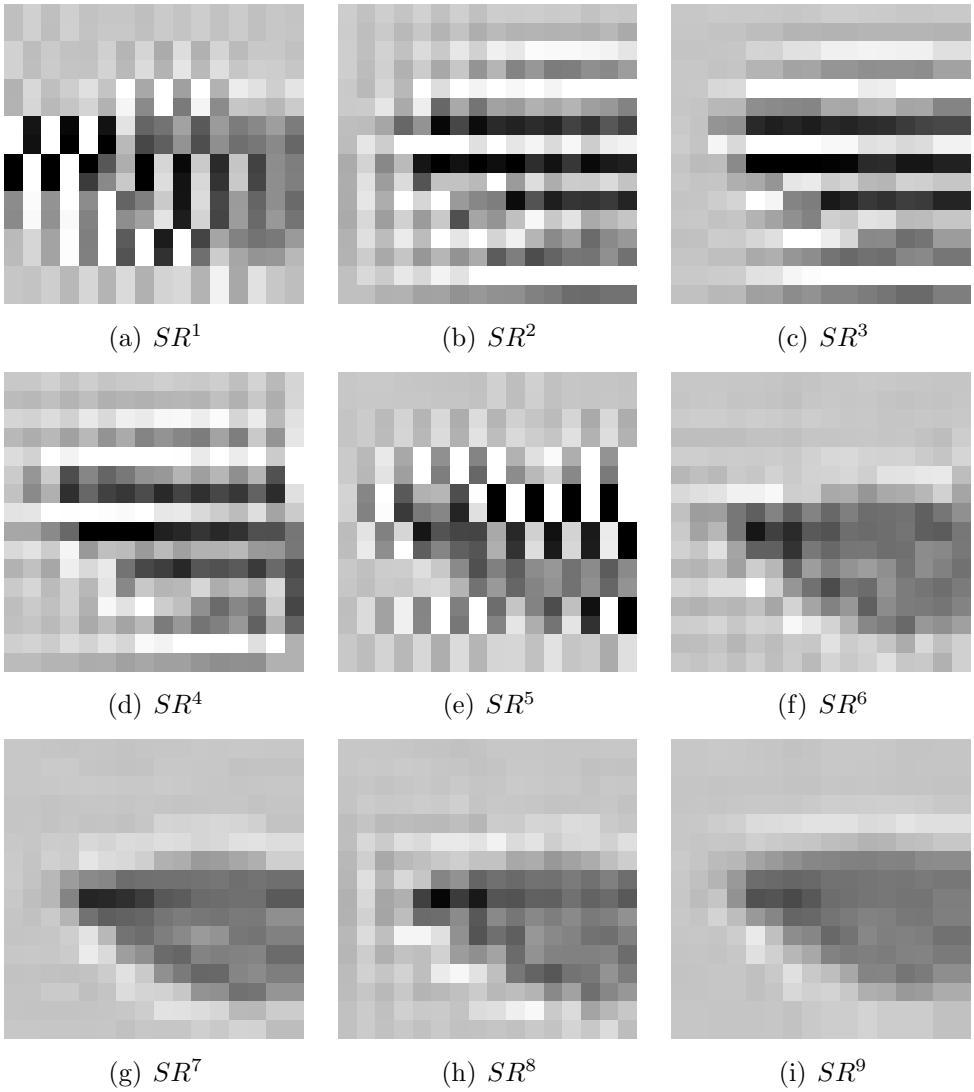
(a) $b_{1,k}$ (b) $b_{0,l}$

Figure 4.3: Example: Two 8×8 blocks shifted by 0.5 pixels

$M^1, ...M^9$ - related to the nine possible halfpel shifts as shown in table 4.1. Note that in this example M^7 is the true underlying motion matrix.
Given these matrices and $F_{ML}(x)$ from equation 4.1 as the function to be minimised there are nine ways to reconstruct a Super-Resolution image. Figure 4.4 shows the resulting SR images $SR^1, ..., SR^9$ which were computed by applying 5 iterations of the Conjugate Gradient algorithm using a bilinear interpolation of $Y_j(l)$ as the initial guess: When comparing the original LR block from figure 4.3 (a) with the nine possible SR images in figure 4.4 it can be seen that SR^7 gives a very accurate reconstruction (the true motion matrix has been chosen), SR^6, SR^8 and SR^9 provide acceptable reconstructions (a motion matrix close to the true motion matrix has been chosen) whereas $SR^1, ...SR^5$ show unacceptable reconstruction results (an inaccurate

M	Corresponds to a shift of
M^1	$(-\frac{1}{2}\|0)$
M^2	$(-\frac{1}{2}\|\frac{1}{2})$
M^3	$(0\|\frac{1}{2})$
M^4	$(\frac{1}{2}\|\frac{1}{2})$
M^5	$(\frac{1}{2}\|0)$
M^6	$(\frac{1}{2}\|-\frac{1}{2})$
M^7	$(0\|-\frac{1}{2})$
M^8	$(-\frac{1}{2}\|-\frac{1}{2})$
M^9	$(0\|0)$

Table 4.1: The nine motion matrices for halfpel shifts



(a) SR^1 (b) SR^2 (c) SR^3

(d) SR^4 (e) SR^5 (f) SR^6

(g) SR^7 (h) SR^8 (i) SR^9

Figure 4.4: Example: The nine resulting SR images, ML approach

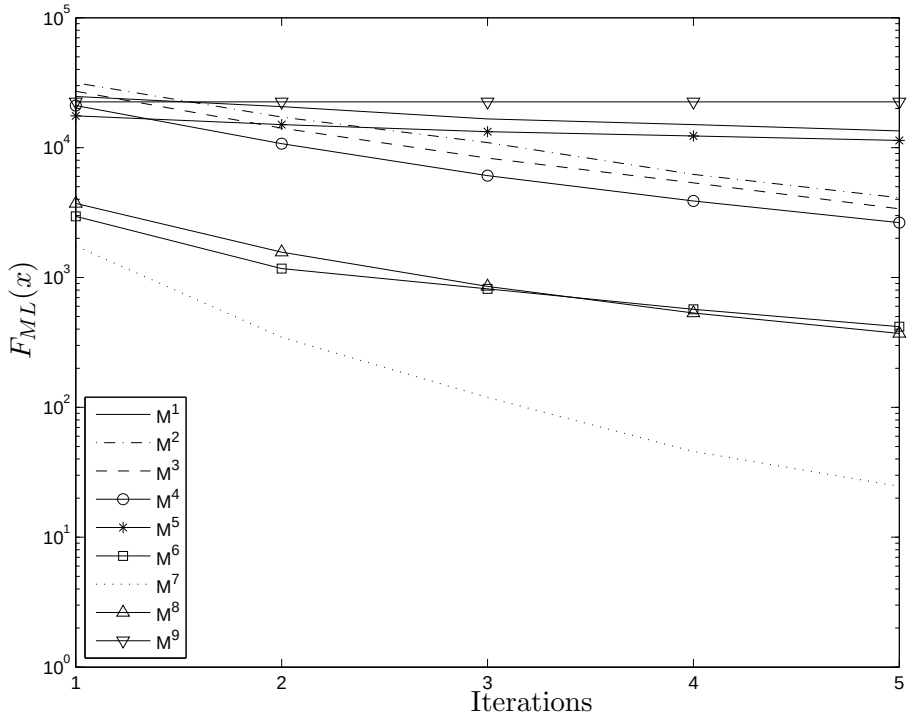


Figure 4.5: Example: Convergence of the cost function $F_{ML}(x)$ for the nine possible SR blocks, Blocksize 8×8

motion matrix has been chosen).

Figure 4.5 shows the decrease of the cost function $F_{ML}(x)$ for all nine possible motion matrices. It can be seen that the true motion matrix (M^7) returns the lowest costs and the best convergence. Given this behaviour the following algorithm for a motion estimator with halfpel accuracy can be formulated:

Iterations	1	2	3	4	5	6	7	8	9	10
% correct decision	31.7	60.0	76.6	79.0	73.5	66.7	62.6	32.0	31.9	31.3
MAD	0.58	0.35	0.25	0.26	0.33	0.43	0.49	0.99	1.00	1.01

Table 4.2: Iterations versus quality of the vectorfield, ML approach

1. Let $\hat{x}^0$ be the interpolated version of $b_{1,k}$, e.g. by using the bilinear interpolation
2. Set $l = 0$
3. While $l \leq L$
   - For all motion matrices M^j with $j = 1, ..., 9$
     - Minimise $F_{ML}(x)$ with respect to x using R iterations of the CG algorithm with $\hat{x}^0$ as the initial guess to yield $\hat{x}_{ML}$, a Maximum Likelihood estimate of x
     - Save the final costs for the l-th block and the j-th motion matrix as $C_{ML}(l,j) = (b_{1,k} - DB\hat{x}_{ML})^T(b_{1,k} - DB\hat{x}_{ML}) + (b_{0,l} - DBM^j\hat{x}_{ML})^T(b_{0,l} - DBM^j\hat{x}_{ML})$
   - $l++$
4. Set $(\tilde{l}, \tilde{j}) = \arg\min_{l,j}\{C_{ML}(l,j)\}$
5. Let $b_{0,\tilde{l}}$ together with motion matrix $M^{\tilde{j}}$ describe the motion vector. Hereby $b_{0,\tilde{l}}$ describes a large-scale motion and $M^{\tilde{j}}$ describes a small-scale motion.

There is one big problem arising when using the maximum likelihood approach: The equation system $A_{ML}x = b$ which has to be solved by the CG algorithm is underdetermined, resulting in a high number of HR images x which all solve the cost function. Thus, after a certain number of iterations every motion matrix M^m (except for M^9) will result in zero costs and the algorithm described above has no meaning anymore. Due to this convergence problem it is desirable to keep the number of iterations small. On the other hand, one loses performance when stopping the optimisation after a small number of iterations, especially when small blocksizes are used and noise is present. This behaviour can be seen in figure 4.6. In this plot the decrease of the cost functions for the nine motion matrices is shown when using a blocksize of 4×4. Again a true motion vector of $(0, -0.5)$ has been chosen. It can be seen that in this case at least 3 iterations would be necessary to clearly identify the true motion.

In table 4.2 the percent of correct motion vectors and the MAD of the estimated vectorfield to the true vectorfield dependent on the number of iterations of the CG algorithm is shown. For this example the 'Coastguard' sequence has been chosen with a simulated displacement of $(0.5, 0.5)$ and a blocksize of 4×4. From the table it can be seen that the optimal number of iterations is around $3 - 4$. A loss in accuracy of the estimated vectorfield is observable when choosing the number of iterations to

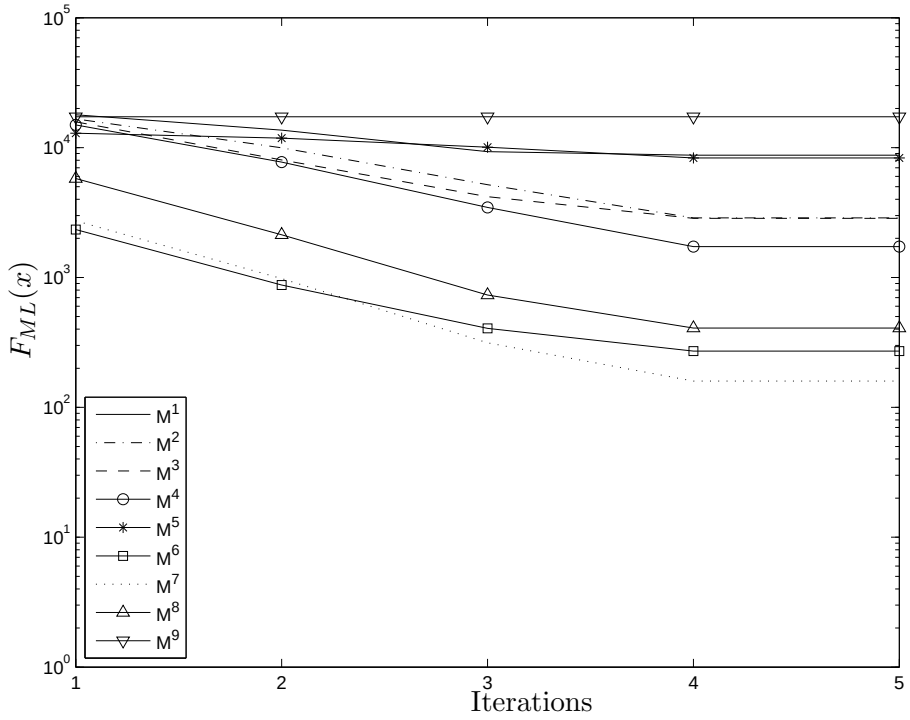


Figure 4.6: Example: Convergence of the cost function $F_{ML}(x)$, for the nine possible SR blocks, Blocksize 4×4

be either less than 3 or larger than 4. This is clearly an unwanted feature. First, it is desirable that the percent of correct decisions increases and the MAD decreases with an increasing number of iterations such that a tradeoff between speed and accuracy can be managed. Secondly, the optimal number of iterations is hard to find in practice.

The reason for this behaviour is the fact that the equation system $A_{ML}(x)x = b$ is underdetermined. Given a sufficient high number of iterations all possible motion matrices solve the cost function. The proposed algorithm (let the motion matrix which results in the lowest costs correspond to the true motion) becomes meaningless when the number of iterations has been chosen to high. By using the MAP approach as depicted in chapter 3.2 this problem is solved by making use of a stochastic regularisation, resulting in the cost function

$$\begin{aligned} F_{MAP,G}(x) = (b_{1,k} - DBx)^T(b_{1,k} - DBx) + \\ (b_{0,l} - DBM^j x)^T(b_{0,l} - DBM^j x) + c_G(x - Nx)^T(x - Nx) \end{aligned} \tag{4.2}$$

with $c_G = \frac{\sigma_\eta^2}{\sigma_G^2}$ for the Gaussian prior or

$$\begin{aligned} F_{MAP,D}(x) = (b_k - DBM_k x)^T(b_k - DBM_k x) + \\ +(b_{f_r} - DBM_l x)^T(b_{f_r} - DBM_l x) + c_D|x - Nx| \end{aligned} \tag{4.3}$$

with $c_D = \frac{\sqrt{2}\sigma_\eta^2}{\sigma_D}$ for the double-exponential prior.

By choosing an appropriate regularisation parameter $c \in \mathbb{R}^+$ it is now possible

Iterations	1	2	3	4	5	6	7	8	9	10
% correct decision	30.4	59.8	78.0	82.1	82.4	81.6	81.9	81.8	81.7	82.1
MAD	0.60	0.36	0.25	0.23	0.23	0.24	0.23	0.23	0.23	0.23

Table 4.3: Iterations versus quality of the vectorfield, MAP approach

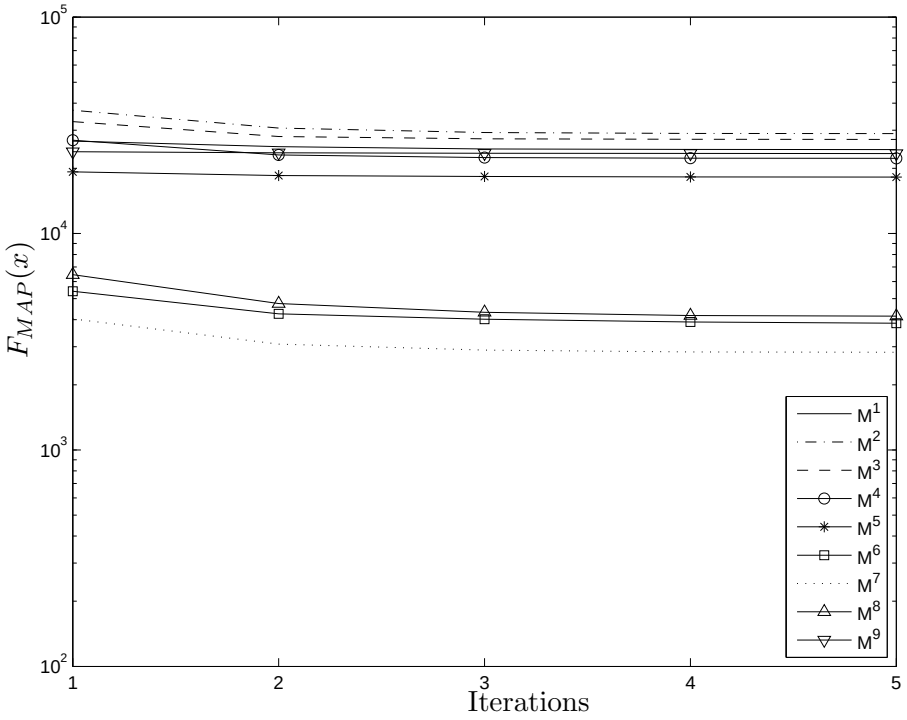
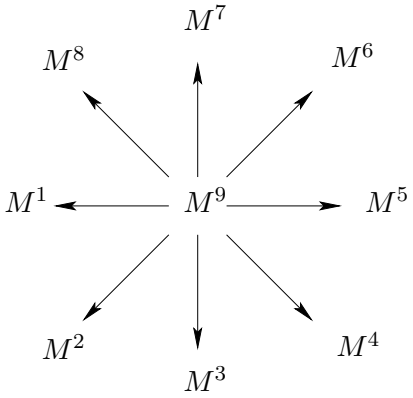
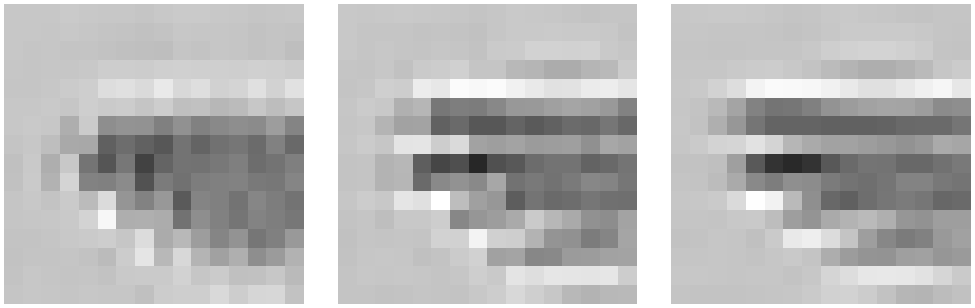


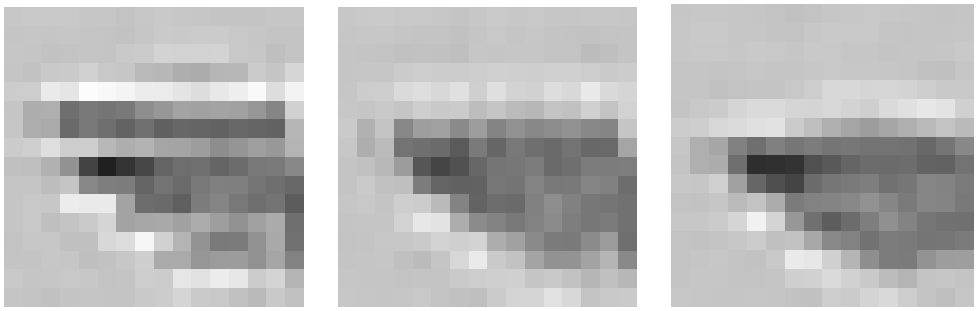
Figure 4.7: Example: Convergence of the cost function $F_{MAP}(x)$ for the nine possible SR blocks, Blocksize 8×8

to control the quality of the estimated vectorfield with the number of iterations. In table 4.3 the same example as described above has been chosen, now with a regularisation parameter $c_G = 0.01$ instead of $c_G = 0$ using the Gaussian prior. The desired property - the percent of correct decisions increases and the MAD decreases with an increasing number of iterations - can be observed.
Recalling the problem of estimating the motion between the blocks shown in figure 4.3 the cost functions for the MAP approach with a Gaussian prior is plotted in figure 4.7. In this plot it can be seen that the SR block which has been constructed assuming the true motion matrix M^7 returns the lowest costs. As opposed to the ML approach, the cost functions in the MAP approach do not converge towards zero as the equation system $A_{MAP}x = b_{MAP}$ is generally overdetermined.
The nine resulting SR blocks after five iterations of the CG algorithm can be seen in figure 4.8. It can be seen that due to the regularisation a solution of $A_{MAP}x = b_{MAP}$ which provides global smoothness is preferred. Similar results can be observed when using the double-exponential prior. One may now formulate the following algorithm for Super-Resolution based motion estimation using the MAP approach:

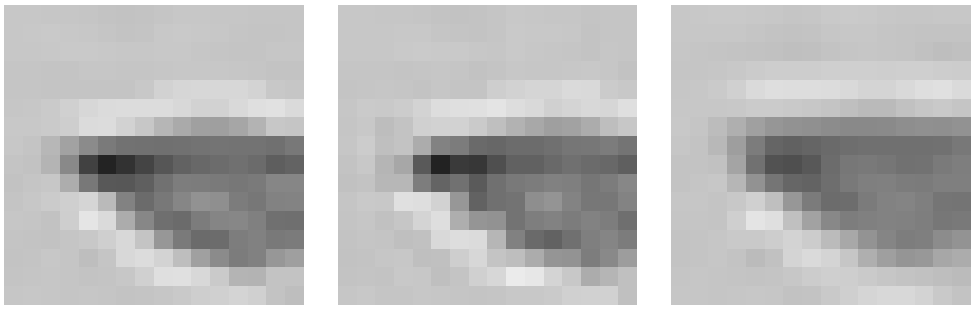




(a) SR^1 (b) SR^2 (c) SR^3



(d) SR^4 (e) SR^5 (f) SR^6



(g) SR^7 (h) SR^8 (i) SR^9

Figure 4.8: Example: The nine resulting SR images, MAP approach

1. Let $\hat{x}^0$ be the interpolated version of $b_{1,k}$, e.g. by using the bilinear interpolation
2. Choose a regularisation parameter $c > 0$
3. Set $l = 0$
4. While $l \leq L$
   - For all motion matrices M^j with $j = 1, ..., 9$
     - Minimise $F_{MAP,G}(x)$ or $F_{MAP,D}(x)$ with respect to x using R iterations of the CG algorithm with $\hat{x}^0$ as the initial guess to yield $\hat{x}_{MAP}$, a MAP estimate of x
     - Save the final costs for the l-th block and the j-th motion matrix as $C_{MAP,G}(l,j) = (b_{1,k} - DB\hat{x}_{MAP,G})^T(b_{1,k} - DB\hat{x}_{MAP,G}) + (b_{0,l} - DBM^j\hat{x}_{MAP,G})^T(b_{0,l} - DBM^j\hat{x}_{MAP,G}) + c_G(\hat{x}_{MAP,G} - N\hat{x}_{MAP,G})^T(\hat{x}_{MAP,G} - N\hat{x}_{MAP,G})$ or $C_{MAP,DE}(l,j) = (b_{1,k} - DB\hat{x}_{MAP,D})^T(b_{1,k} - DB\hat{x}_{MAP,D}) + (b_{0,l} - DBM^j\hat{x}_{MAP,D})^T(b_{0,l} - DBM^j\hat{x}_{MAP,D}) + c_D|\hat{x}_{MAP,D} - N\hat{x}_{MAP,D}|$ depending on whether one uses the Gaussian or the double-exponential prior.
   - $l++$
5. Set $(\tilde{l}, \tilde{j}) = \arg\min_{l,j}\{C_{MAP}(l,j)\}$
6. Let $b_{0,\tilde{l}}$ together with motion matrix $M^{\tilde{j}}$ describe the motion vector. Hereby $b_{0,\tilde{l}}$ describes a large-scale motion and $M^{\tilde{j}}$ describes a small-scale motion.

4.2.1 Computational complexity

A short review about the computational complexity of the SR-based motion estimator compared with classical motion estimation schemes is given here. The Matlab function 'cputime' will be used as a measure for computational complexity. Given a fixed blocksize of 4×4, an image size of 176×144 pixels and a search range of 4 the complexity of the SR-based motion estimator is dependent on the number of iterations of the CG algorithm and the choice of the prior (Gaussian or Double-Exponential). The computation time of the SR-based motion estimator will be plotted relative to the computation time of the SAD blockmatcher which is the fastest motion estimation scheme presented in this thesis.

In figure 4.9 the computational complexity of the SR-based motion estimator with Gaussian and Double-Exponential prior are plotted. In can be seen that there is a linear dependence between the number of iterations of the CG algorithm and the computation time. Furthermore due to the need for nonlinear optimisation algorithms the usage of the Double-Exponential prior leads to more computational complexity.

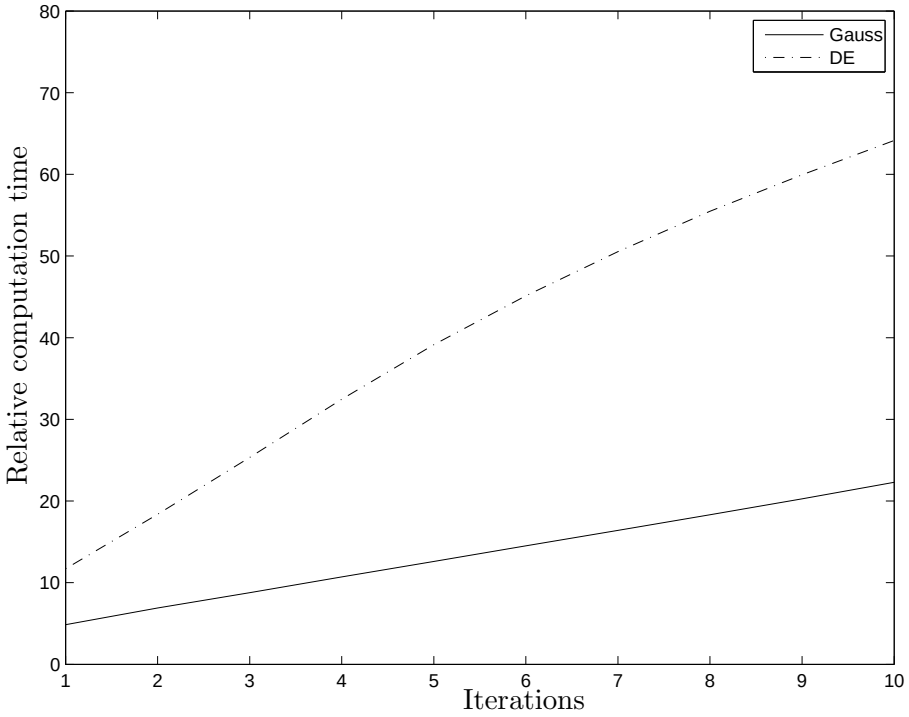


Figure 4.9: Computational complexity

It was shown in the last section that 3 iterations are sufficient for the motion estimator to identify the true motion in most cases. Using more than 3 iterations will only lead to negligible improvements of the estimated vectorfield. Given 3 iterations of the CG algorithm as the 'default' option it can be seen that the SR-based motion estimator with a Gaussian prior increases computational complexity by a factor of about 10 whereas computational complexity is increased by a factor of about 25 when using the Double-Exponential prior.

4.3 Simulation results

In this section simulation results comparing the Super-Resolution-based motion estimator with existing motion estimators are shown. The estimators are compared under different scenarios including aliasing and noise effects. For the simulation setup as described in chapter 2 the luminance part of the first frames of the following YUV sequences will be used:

- **Coastguard**, 352×288 pixels
- **Mobile**, 352×288 pixels

The following parameters will be used for simulating the downsampling process

- **Downsampling factor:** $p = 2$
- **Shift:** $(\frac{1}{2}, \frac{1}{2})$
- **Number of iterations:** 3

Furthermore the motion estimator may take one of the following options:

- **Regularisation parameter:**
  $c \in \{0, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5\}$

In the following, some representative plots are shown, exploring the effect of aliasing, noise, regularisation parameter and the MAP prior on the quality of the estimated vectorfield. The Super-Resolution based motion estimator will be compared with a classical block based motion estimator using the SAD cost function and the 6 tap FIR interpolation filter. The complete simulation results of all test scenarios can be found in the appendix.
The motion estimator has a halfpel accuracy, the searchrange is fixed to $s = 2p$. As an initial guess $\hat{x}^0$, the Super-Resolution based motion estimator uses an interpolated version of b_k (the block for which a motion vector is wanted) using the 6 tap FIR interpolation filter presented in chapter 2.

4.3.1 The effect of the regularisation on motion estimates

As shown in the previous section and in chapter 3, regularisation is an important issue for both image quality in Super Resolution and reliability of the motion estimates. A double-exponential prior provides a more accurate image model, leading to better estimated SR images compared with the Gaussian prior. It is expected that similarly for motion estimation a gain in performance is observable when using a better image model. The influence of the regularisation parameter c when using the Gaussian and Double-Exponential prior will be examined here. In figure 4.10 the performance of the SR based motion estimator using the 'Mobile' sequence for different regularisation parameters ($c = 0, ..., 0.5$) is shown for 3 iterations of the CG algorithm. It can be seen that the Double-Exponential prior leads to by far better results. Especially when a large regularisation parameter is used (i.e. when the image model dominates the cost function) the SR-based motion estimator with a Gaussian prior rapidly loses performance. Oversmoothed solutions far away from the image characteristics are found leading unreliable motion estimates.
A similar results can be found for the 'Coastguard' sequence in figure 4.11. However the pdf of $(x - Nx)$ for the 'Coastguard' sequence is closer to a Gaussian shape than the one for the 'Mobile' sequence. As long as the regularisation parameter is kept at a small level the Gaussian prior leads to reliable motion estimates.

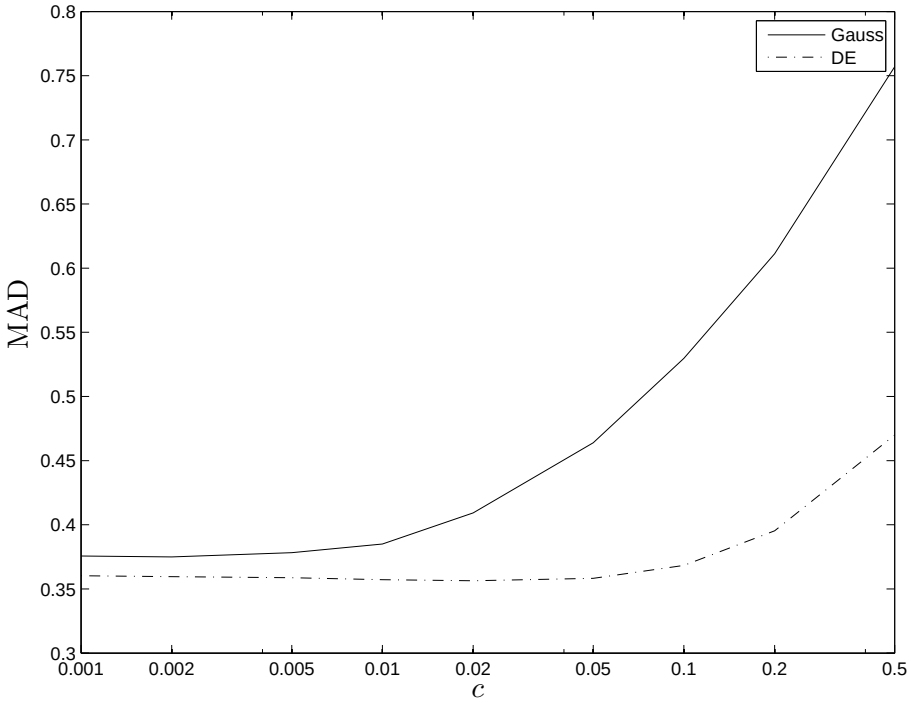


Figure 4.10: Simulation result: Effect of the regularisation on the MAD of motion estimates, 'Mobile' sequence

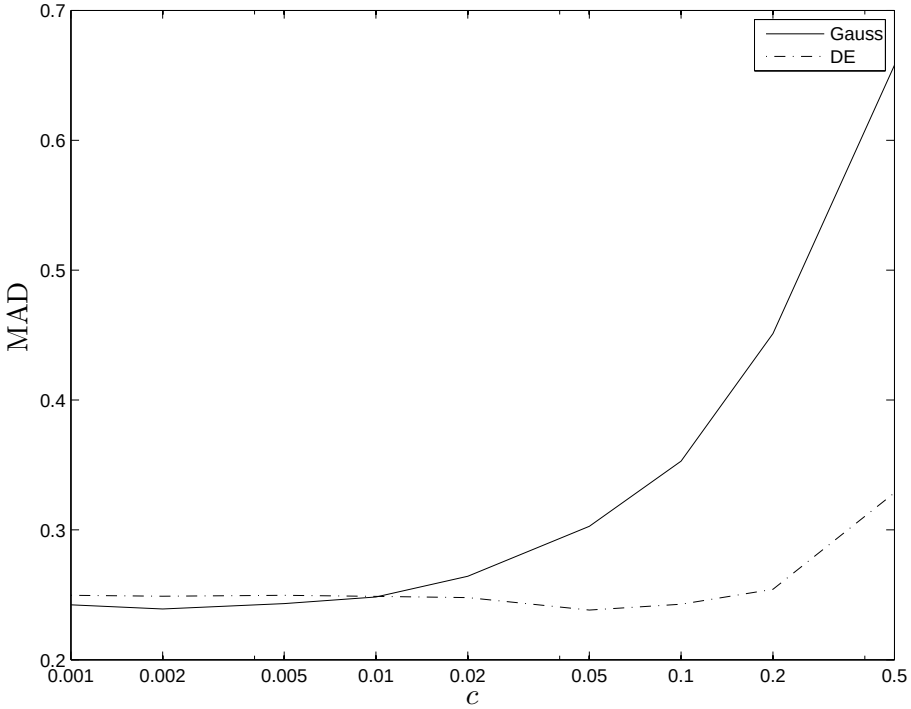


Figure 4.11: Simulation result: Effect of the regularisation on the MAD of motion estimates, 'Coastguard' sequence

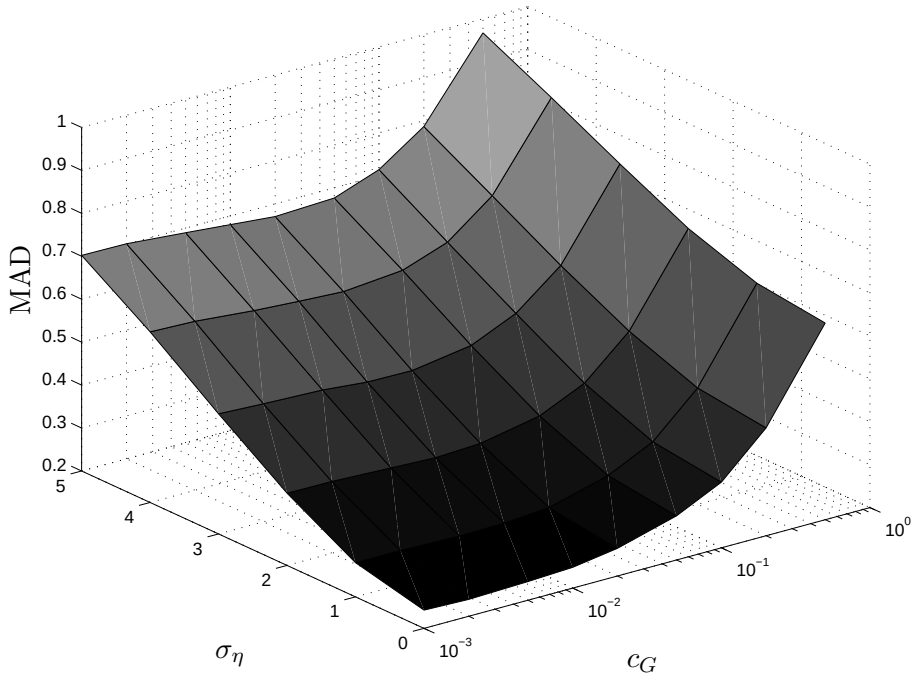


Figure 4.12: Simulation result: Effect of noise on the MAD of motion estimates,'Coastguard' sequence, Gaussian prior

4.3.2 The effect of noise on motion estimates – Gaussian prior

In chapter 3 the destructive effect of noise on the super-resolved image was shown. It is expected that noise will have a similar effect on the estimated vectorfield. The noise is modeled as being Gaussian distributed with zero mean and variance σ_η^2, simulation results have been averaged over 100 Monte Carlo simulations.

In figure 4.12 the effect of noise on motion estimation is shown for the 'Coastguard' sequence with 3 iterations. It can be observed that a small gain in performance could be achieved by choosing c adaptive to σ. For a small value of σ, only a small regularisation parameter is necessary ($c < 0.01$) whereas for bigger values of σ a higher regularisation should be used.

In figure 4.13 the simulation results for the 'Mobile' sequence are shown. It can be seen that there would be little gain in performance when choosing c adaptive to σ. As the GMRF is a very inaccurate model for the 'Mobile' sequence the best performance can be achieved when choosing a very small regularisation parameter, irrespective of the noise level.

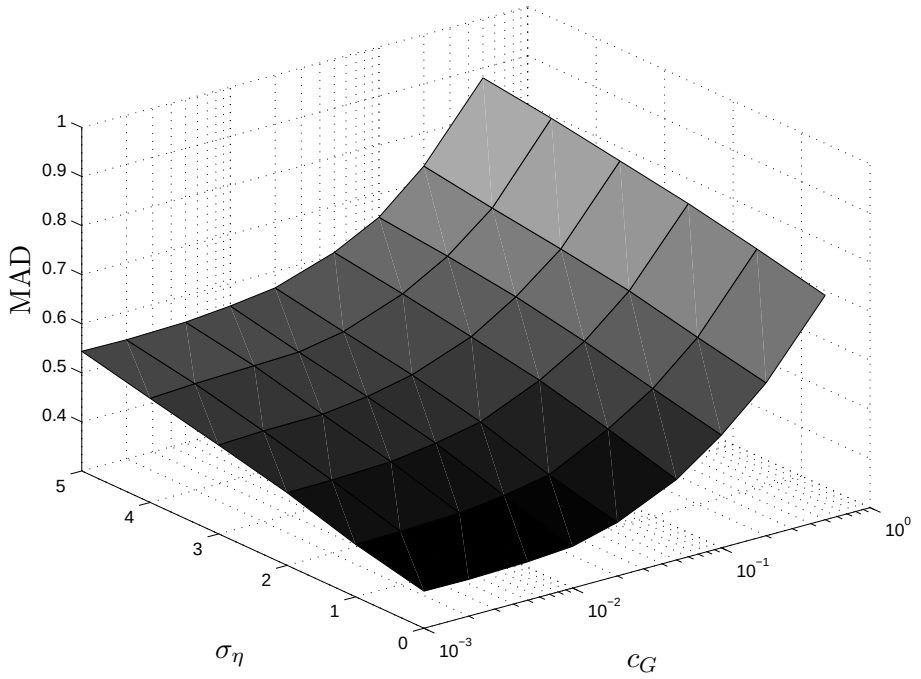


Figure 4.13: Simulation result: Effect of noise on the MAD of motion estimates, 'Mobile' sequence, Gaussian prior

4.3.3 The effect of noise on motion estimates – Double-Exponential prior

In figures 4.14 and 4.15 the quality of the estimated vectorfield for different noise levels and regularisation parameters is shown when using a Double-Exponential prior. A similar conclusion as in the previous section can be drawn: When using the Double-Exponential prior the quality of the estimated vectorfield is less sensitive to changes from c. Furthermore a slight improvement is visible when choosing a small regularisation parameter for the noise-free case ($\sigma_\eta = 0$) and increasing c as σ_η increases.

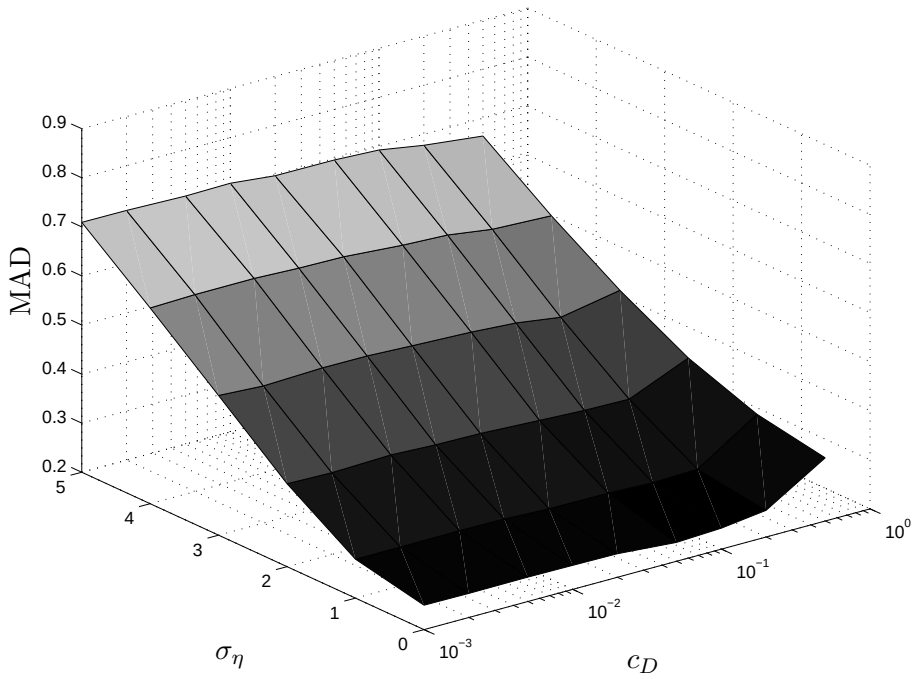


Figure 4.14: Simulation result: Effect of noise on the MAD of motion estimates, 'Coastguard' sequence, Double-Exponential prior

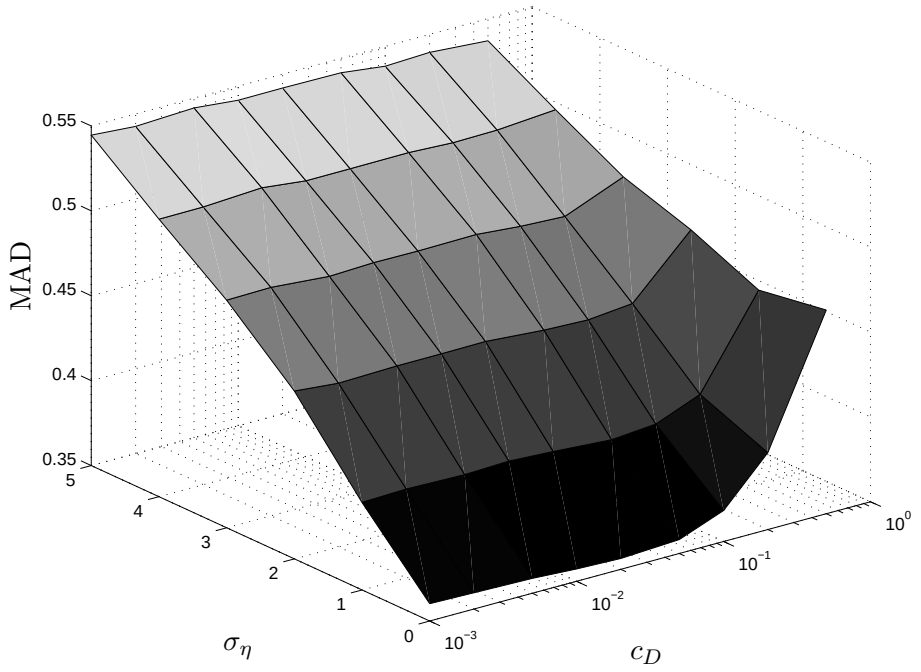


Figure 4.15: Simulation result: Effect of noise on the MAD of motion estimates, 'Mobile' sequence, Double-Exponential prior

4.4 Conclusion

In this chapter the SR-based motion estimator has been presented. Its performance has been shown in numerous simulations showing the effect of regularisation and noise on the motion estimates. The following conclusions can be drawn:

- The SR-based motion estimator is less sensitive to aliasing effects than classical motion estimation schemes based on the intensity conservation principle.
- Regularisation is necessary to manage the ill-posed nature of the optimisation problem and to obtain stable solutions.
- Using the Double-Exponential prior leads to improved performance for motion estimation compared with the Gaussian prior as it fits better to most image statistics.
- The Double-Exponential prior requires non-linear optimisation which results in a high computational complexity. When using the Gaussian prior the optimisation problem is linear and can effectively be solved by e.g. the linear Conjugate Gradient algorithm.
- As the CG algorithm converges quickly to a minimum of the cost function 3 iterations are sufficient to identify the true motion.

Chapter 5

Conclusions

The work done in this thesis can be divided into three parts: In chapter 2 classical motion estimation schemes are reviewed. Their performance under non-ideal circumstances has been examined. It was shown that a high loss of performance can be observed when the assumption of intensity conservation is not fulfilled (due to aliasing and noise).
In chapter 3 the concept of Super-Resolution using stochastic regularisation has been treated. The effect of different parameters such as the number of iterations of the optimisation algorithm and regularisation parameters on the super-resolved image has been examined. Two priors have been used for regularisation: The Gaussian and the Double-Exponential prior. The latter one shows superior results at the cost of a higher computational complexity.
In chapter 4 a new motion estimator based on the Super-Resolution concept has been presented. It does not rely on the principle of intensity conservation and offers superior performance when aliasing is present compared to the classical motion estimation schemes from chapter 2 at the cost of a higher computational complexity. As for the Super-Resolution concept regularisation plays an important role to stabilise the optimisation problem and finally lead to reliable motion estimates.
The new motion estimator is expected to achieve superior performance in applications where high-precision and highly reliable estimates of motion are of crucial importance.

Further work based on this thesis could be done in the following areas:

- The point spread function (PSF) of the camera has been modeled as an averaging function which is time- and spaceinvariant. However in practice the PSF may have other shapes (e.g. a Gaussian shape) and as it depends on the focusing it may at least be spacevariant. This could be included in the new motion estimator by a prior estimation of the PSF [16].
- The noise in the image acquisition process has been assumed to be additive and i.i.d. Gaussian distributed. This assumption is often made due to the

central limit theorem and the consequential simple form of the optimisation problem. However CCD camera noise is known to be intensity-dependent [22, 7]. Furthermore there are scenarios in which the noise pdf deviates from a Gaussian shape, e.g. when salt'n'pepper noise is present. A motion estimation scheme which incorporates knowledge about the pdf of the noise (which may be space- and timevariant) is assumed to have an improved performance.

- The regularisation parameter c and the penalty function (or the image model respectively) have been chosen as being static. However if knowledge of the noise pdf and statistics of the image model are present it would be possible to design an adaptive regularisation term.

- In this thesis only translational motion has been considered - which is well motivated if small block sizes are chosen and if the time difference between images is short. However more general motion models including rotation and warping can be incorporated in the presented motion estimator by an additional set of motion matrices.

- The image observation model used in this thesis does not account for effects such as zooming or changes in the brightness between subsequent images. However when knowledge about additional effects is available (e.g. by a prior estimate of zooming [11, 10] and an estimate of the change in brightness via histogram comparison [6]) this knowledge can be incorporated in the image model by additional matrices without changing the overall procedure of SR-based motion estimation.

- Only the GMRF and DEMRF image models have been treated in this thesis. However, there are other models, such as the Huber Markov Random Field (HMRF) image model which have shown good performance in the area of Super-Resolution [18, 21]. A preliminary study about HMRF image models for use in motion analysis has been presented in [3]. Further studies are necessary to examine the effect of different penalty functions using the HMRF image model on motion estimation.

Appendix A

Motion Analysis - Simulation Results

	Nearest Neighbour						Bilinear					
	σ_η						σ_η					
	0	1	2	3	4	5	0	1	2	3	4	5
SAD	1.23	1.24	1.27	1.30	1.35	1.38	0.68	0.73	0.80	0.87	0.96	1.04
SSD	1.22	1.22	1.26	1.30	1.34	1.38	0.66	0.69	0.76	0.84	0.93	1.01
SATD	1.23	1.25	1.28	1.32	1.36	1.41	0.75	0.79	0.87	0.96	1.04	1.12
SSTD	1.22	1.23	1.25	1.30	1.34	1.38	0.66	0.69	0.76	0.84	0.93	1.01
	Bicubic						6 tap					
	σ_η						σ_η					
	0	1	2	3	4	5	0	1	2	3	4	5
SAD	0.55	0.58	0.65	0.74	0.84	0.94	0.48	0.51	0.58	0.67	0.77	0.87
SSD	0.51	0.54	0.62	0.71	0.82	0.91	0.44	0.47	0.55	0.64	0.75	0.84
SATD	0.55	0.59	0.69	0.80	0.91	1.00	0.45	0.49	0.58	0.68	0.79	0.89
SSTD	0.51	0.54	0.62	0.71	0.82	0.91	0.44	0.47	0.54	0.64	0.74	0.84

Table A.1: Simulation results: MAD, 'Coastguard' sequence

	Nearest Neighbour						Bilinear					
	σ_η						σ_η					
	0	1	2	3	4	5	0	1	2	3	4	5
SAD	1.05	1.10	1.14	1.18	1.20	1.23	0.68	0.73	0.78	0.82	0.85	0.88
SSD	1.07	1.11	1.14	1.17	1.19	1.22	0.66	0.71	0.75	0.78	0.80	0.83
SATD	1.11	1.15	1.18	1.21	1.23	1.24	0.74	0.79	0.84	0.87	0.90	0.93
SSTD	1.07	1.11	1.14	1.17	1.20	1.21	0.67	0.71	0.75	0.78	0.80	0.83
	Bicubic						6 tap					
	σ_η						σ_η					
	0	1	2	3	4	5	0	1	2	3	4	5
SAD	0.59	0.64	0.69	0.73	0.76	0.79	0.52	0.56	0.61	0.64	0.67	0.70
SSD	0.57	0.61	0.65	0.68	0.72	0.74	0.49	0.52	0.56	0.59	0.62	0.65
SATD	0.62	0.66	0.71	0.75	0.78	0.81	0.51	0.55	0.59	0.63	0.66	0.69
SSTD	0.57	0.61	0.65	0.69	0.72	0.75	0.49	0.53	0.56	0.59	0.62	0.65

Table A.2: Simulation results: MAD, 'Mobile' sequence

Appendix B

Super Resolution - Simulation Results

	σ_η					
	0	1	2	3	4	5
c	SSIM					
0	0.96726	0.96035	0.94164	0.91226	0.87466	0.83116
0.01	0.96723	0.96063	0.94266	0.91435	0.87822	0.83626
0.02	0.96704	0.96067	0.94336	0.91607	0.88078	0.84037
0.03	0.96674	0.96062	0.9439	0.91745	0.88351	0.84419
0.04	0.96637	0.96036	0.94418	0.91859	0.88544	0.84743
0.05	0.96582	0.96008	0.94434	0.91954	0.88729	0.85014
0.06	0.96518	0.95963	0.94439	0.92023	0.88892	0.85259
0.07	0.96448	0.95909	0.94427	0.92082	0.89042	0.85489
0.08	0.9637	0.95852	0.9441	0.92125	0.89148	0.85673
0.09	0.96286	0.95778	0.94381	0.92159	0.89258	0.85858
0.1	0.96196	0.95703	0.94341	0.92184	0.89344	0.86034
0.2	0.95031	0.94642	0.93571	0.91861	0.89592	0.86877
0.3	0.93758	0.93434	0.92542	0.91098	0.89177	0.86854
0.4	0.92651	0.92362	0.91568	0.90312	0.88618	0.86569
0.5	0.91684	0.91418	0.90702	0.89536	0.8801	0.86151
0.6	0.90782	0.90543	0.8988	0.88821	0.87422	0.8569
0.7	0.89941	0.89718	0.89101	0.8811	0.86812	0.85236
0.8	0.89136	0.88925	0.88348	0.87424	0.86192	0.84727
0.9	0.88349	0.88154	0.87615	0.86769	0.85622	0.84253
1	0.87599	0.87408	0.86901	0.86101	0.85035	0.83735
	PSNR					
0	36.9145	36.3589	35.0862	33.5181	31.9498	30.484
0.01	36.9467	36.4154	35.1804	33.6532	32.1151	30.6661
0.02	36.9535	36.4382	35.2447	33.7577	32.2398	30.825
0.03	36.9399	36.4469	35.2929	33.8424	32.3673	30.9661
0.04	36.9095	36.4341	35.3171	33.9076	32.4521	31.0922
0.05	36.8655	36.4073	35.3304	33.9629	32.5407	31.1941
0.06	36.8003	36.3661	35.3322	33.9983	32.6131	31.2861
0.07	36.7284	36.313	35.3196	34.0267	32.673	31.3754
0.08	36.6462	36.2531	35.2936	34.0445	32.7226	31.4403
0.09	36.5588	36.1802	35.2641	34.048	32.7666	31.5128
0.1	36.4642	36.1033	35.2215	34.0538	32.7962	31.5739
0.2	35.3581	35.1341	34.5614	33.7459	32.8081	31.8268
0.3	34.3748	34.2281	33.833	33.2448	32.5278	31.7454
0.4	33.6596	33.5448	33.2416	32.7869	32.2126	31.5682
0.5	33.1097	33.0151	32.7712	32.3899	31.9098	31.3634
0.6	32.6538	32.5781	32.372	32.0514	31.6435	31.1612
0.7	32.2711	32.2051	32.0281	31.7503	31.3943	30.9771
0.8	31.9382	31.8814	31.7264	31.4826	31.164	30.7947
0.9	31.6445	31.593	31.4557	31.2407	30.9558	30.6239
1	31.375	31.3293	31.2063	31.013	30.7596	30.4566

Table B.1: Simulation results: Gaussian prior, SSIM and PSNR, 'Coastguard' sequence

	σ_η					
	0	1	2	3	4	5
c	SSIM					
0	0.96828	0.96128	0.94255	0.91295	0.87539	0.83227
0.01	0.96833	0.96138	0.94274	0.91342	0.87591	0.83281
0.02	0.96836	0.96147	0.94299	0.914	0.87644	0.83356
0.03	0.96838	0.96155	0.94324	0.91431	0.87704	0.83417
0.04	0.96839	0.96163	0.94339	0.91469	0.8776	0.83461
0.05	0.96839	0.96167	0.94355	0.91492	0.87796	0.83511
0.06	0.96837	0.96169	0.94375	0.91522	0.87855	0.8362
0.07	0.96834	0.96175	0.9439	0.91562	0.87898	0.83679
0.08	0.9683	0.96176	0.94402	0.91588	0.87942	0.83723
0.09	0.96825	0.96176	0.94417	0.91621	0.87991	0.83778
0.1	0.9682	0.96175	0.94428	0.91654	0.88045	0.83855
0.2	0.96721	0.96125	0.94498	0.9187	0.88406	0.84351
0.3	0.96562	0.96009	0.94479	0.91999	0.88658	0.84749
0.4	0.9636	0.95841	0.94414	0.92065	0.88887	0.8507
0.5	0.9612	0.95636	0.94301	0.92076	0.89031	0.85377
0.6	0.95838	0.95393	0.94144	0.92042	0.89146	0.85591
0.7	0.94549	0.94228	0.93276	0.91844	0.89236	0.85832
0.8	0.93489	0.93194	0.92323	0.90858	0.88784	0.86057
0.9	0.93044	0.92805	0.91647	0.89819	0.87971	0.85628
1	0.90117	0.89894	0.89304	0.88486	0.8781	0.84853
	PSNR					
0	37.1175	36.5295	35.2148	33.6042	32.0042	30.5264
0.01	37.1252	36.5402	35.2284	33.6271	32.0309	30.5441
0.02	37.1319	36.5498	35.2451	33.6516	32.0489	30.5712
0.03	37.1374	36.5589	35.2577	33.6667	32.0714	30.5879
0.04	37.142	36.5695	35.2714	33.6856	32.0924	30.6078
0.05	37.1457	36.5742	35.2842	33.6986	32.1094	30.6273
0.06	37.1482	36.5799	35.2973	33.7156	32.1325	30.6601
0.07	37.1497	36.5865	35.3068	33.7345	32.1524	30.675
0.08	37.1503	36.5909	35.3171	33.7471	32.1667	30.6924
0.09	37.15	36.5946	35.3282	33.7646	32.1857	30.7116
0.1	37.1488	36.595	35.3388	33.783	32.2094	30.7339
0.2	37.0975	36.5824	35.3961	33.8966	32.3567	30.8992
0.3	36.9861	36.5124	35.3982	33.9677	32.4654	31.04
0.4	36.8279	36.3945	35.3629	34.0074	32.5588	31.1554
0.5	36.6271	36.2379	35.2919	34.0163	32.6215	31.2642
0.6	36.3808	36.0395	35.1893	33.9993	32.6761	31.3442
0.7	35.1286	34.9585	34.4703	33.8573	32.7145	31.4359
0.8	34.3548	34.2146	33.8324	33.235	32.4468	31.5075
0.9	33.9861	33.8975	33.3902	32.693	32.0734	31.3314
1	32.5184	32.4471	32.2631	32.0374	31.9442	31.0401

Table B.2: Simulation results: DE prior, SSIM and PSNR, 'Coastguard' sequence

	σ_η					
	0	1	2	3	4	5
c	SSIM					
0	0.94679	0.93937	0.91817	0.88788	0.85365	0.81902
0.01	0.94913	0.94225	0.92249	0.89375	0.86128	0.82741
0.02	0.95063	0.94415	0.9252	0.89754	0.86628	0.83352
0.03	0.95119	0.945	0.92701	0.90067	0.87001	0.83851
0.04	0.95107	0.94512	0.92805	0.90285	0.87344	0.84268
0.05	0.95045	0.94479	0.9286	0.90457	0.87648	0.84648
0.06	0.94951	0.94411	0.9288	0.90588	0.87863	0.84957
0.07	0.94832	0.94319	0.92867	0.90671	0.8806	0.85226
0.08	0.94704	0.94209	0.92825	0.90719	0.88206	0.85479
0.09	0.94568	0.94097	0.9277	0.90751	0.88287	0.85651
0.1	0.94429	0.93969	0.92692	0.90748	0.88385	0.85774
0.2	0.92935	0.92586	0.91632	0.90148	0.88298	0.86185
0.3	0.91429	0.91142	0.90365	0.89152	0.87592	0.85794
0.4	0.90095	0.89851	0.89175	0.88118	0.86743	0.85153
0.5	0.88924	0.88701	0.88096	0.87141	0.859	0.84474
0.6	0.93731	0.93095	0.92006	0.90495	0.88399	0.85735
0.7	0.92413	0.92148	0.91843	0.90295	0.88241	0.8589
0.8	0.90908	0.90683	0.90271	0.89433	0.88097	0.85879
0.9	0.89444	0.89247	0.88873	0.8814	0.87072	0.85446
1	0.88319	0.88131	0.87671	0.8696	0.85947	0.8461
	PSNR					
0	28.4024	28.3118	28.0655	27.6807	27.1904	26.6257
0.01	28.5023	28.4161	28.1763	27.8009	27.3267	26.7758
0.02	28.5732	28.4884	28.2538	27.8921	27.4255	26.8901
0.03	28.6191	28.5361	28.3099	27.9521	27.5013	26.9765
0.04	28.6433	28.562	28.34	27.995	27.5531	27.0402
0.05	28.6453	28.5655	28.3526	28.019	27.5887	27.0881
0.06	28.6241	28.5493	28.3453	28.0191	27.6018	27.1245
0.07	28.5873	28.5137	28.3142	28.0015	27.6009	27.1327
0.08	28.5323	28.4618	28.2732	27.9678	27.5854	27.1295
0.09	28.4656	28.3961	28.2154	27.9266	27.5574	27.1202
0.1	28.3853	28.3211	28.1484	27.8753	27.519	27.0967
0.2	27.4435	27.405	27.2988	27.1255	26.8972	26.6144
0.3	26.6402	26.6141	26.5414	26.4214	26.2607	26.0602
0.4	26.0341	26.0147	25.959	25.87	25.745	25.5892
0.5	25.5799	25.5626	25.5184	25.4434	25.3412	25.2149
0.6	25.211	25.1967	25.1585	25.0958	25.0086	24.9009
0.7	24.8944	24.8816	24.8489	24.7945	24.7203	24.6255
0.8	24.6159	24.6047	24.5748	24.528	24.4621	24.3782
0.9	24.3646	24.355	24.3298	24.286	24.2288	24.1538
1	24.1375	24.1289	24.105	24.0676	24.014	23.9474

Table B.3: Simulation results: Gaussian prior, SSIM and PSNR, 'Mobile' sequence

	σ_η					
	0	1	2	3	4	5
c	SSIM					
0	0.94767	0.93932	0.91832	0.88848	0.85481	0.81978
0.01	0.94813	0.93995	0.91908	0.88953	0.85533	0.82067
0.02	0.94854	0.94059	0.92	0.89038	0.85647	0.82179
0.03	0.94891	0.94116	0.9207	0.89143	0.85809	0.82295
0.04	0.94924	0.94165	0.92148	0.89239	0.85857	0.82339
0.05	0.94955	0.94213	0.92212	0.89288	0.85972	0.82436
0.06	0.94983	0.94256	0.92282	0.89396	0.86012	0.82548
0.07	0.9501	0.94296	0.92342	0.89474	0.86117	0.82676
0.08	0.95033	0.94333	0.92404	0.89559	0.86213	0.82743
0.09	0.95055	0.94367	0.92459	0.89615	0.86285	0.82794
0.1	0.95074	0.94398	0.92501	0.89684	0.86384	0.829
0.2	0.95165	0.946	0.9291	0.90247	0.87025	0.83564
0.3	0.95118	0.94612	0.93138	0.90646	0.87519	0.84128
0.4	0.94667	0.94281	0.93099	0.90958	0.88007	0.84669
0.5	0.93601	0.93291	0.92425	0.90815	0.88427	0.8543
0.6	0.93731	0.93095	0.92006	0.90495	0.88399	0.85735
0.7	0.92413	0.92148	0.91843	0.90295	0.88241	0.8589
0.8	0.90908	0.90683	0.90271	0.89433	0.88097	0.85879
0.9	0.89444	0.89247	0.88873	0.8814	0.87072	0.85446
1	0.88319	0.88131	0.87671	0.8696	0.85947	0.8461
	PSNR					
0	28.3722	28.2805	28.0345	27.6497	27.1634	26.5931
0.01	28.3795	28.2874	28.0446	27.6619	27.1726	26.6057
0.02	28.3866	28.2955	28.0522	27.6681	27.1804	26.6153
0.03	28.3935	28.3033	28.0591	27.6792	27.1899	26.6271
0.04	28.4002	28.3104	28.0689	27.6882	27.2018	26.6397
0.05	28.4067	28.3169	28.076	27.6968	27.2119	26.6536
0.06	28.4129	28.3244	28.083	27.7064	27.2211	26.6605
0.07	28.419	28.3307	28.092	27.7129	27.2303	26.6697
0.08	28.4248	28.3366	28.0966	27.7215	27.2416	26.6805
0.09	28.4304	28.3427	28.1046	27.7284	27.2469	26.6906
0.1	28.4357	28.3485	28.112	27.7372	27.2565	26.7008
0.2	28.4757	28.393	28.1653	27.804	27.3337	26.7858
0.3	28.4906	28.4141	28.1991	27.8497	27.3916	26.8558
0.4	28.4815	28.4095	28.2092	27.8738	27.4297	26.906
0.5	28.4532	28.3838	28.1972	27.8829	27.4585	26.9465
0.6	28.2978	28.3096	28.1661	27.8698	27.4663	26.9763
0.7	27.9387	27.9129	27.8949	27.8263	27.4626	26.9917
0.8	27.6692	27.6482	27.5699	27.4371	27.2742	26.9887
0.9	27.4688	27.4547	27.3615	27.2154	27.0086	26.7445
1	27.331	27.3215	27.2262	27.0551	26.8366	26.5757

Table B.4: Simulation results: DE prior, SSIM and PSNR, 'Mobile' sequence

Appendix C

SR-based Motion Estimator - Simulation Results

	σ_η					
c	0	1	2	3	4	5
0	0.25	0.28	0.37	0.49	0.6	0.71
0.001	0.24	0.28	0.37	0.48	0.6	0.7
0.002	0.24	0.28	0.37	0.47	0.59	0.7
0.005	0.24	0.28	0.36	0.47	0.58	0.69
0.01	0.25	0.28	0.36	0.46	0.57	0.68
0.02	0.26	0.29	0.36	0.46	0.57	0.67
0.05	0.3	0.32	0.39	0.48	0.58	0.68
0.1	0.35	0.37	0.44	0.53	0.62	0.71
0.2	0.45	0.47	0.53	0.61	0.7	0.79
0.5	0.66	0.68	0.73	0.81	0.89	0.97

Table C.1: Simulation results: MAD, Gaussian prior, 'Coastguard' sequence

	σ_η					
c	0	1	2	3	4	5
0	0.38	0.4	0.45	0.48	0.51	0.54
0.001	0.38	0.41	0.45	0.48	0.51	0.54
0.002	0.37	0.41	0.45	0.49	0.52	0.54
0.005	0.38	0.41	0.45	0.49	0.52	0.55
0.01	0.39	0.42	0.47	0.5	0.52	0.55
0.02	0.41	0.44	0.49	0.52	0.54	0.57
0.05	0.46	0.5	0.54	0.56	0.59	0.61
0.1	0.53	0.56	0.59	0.62	0.64	0.65
0.2	0.61	0.64	0.67	0.69	0.71	0.73
0.5	0.76	0.79	0.82	0.84	0.86	0.88

Table C.2: Simulation results: MAD, Gaussian prior, 'Mobile' sequence

	σ_η					
c	0	1	2	3	4	5
0	0.25	0.28	0.37	0.49	0.6	0.71
0.001	0.25	0.28	0.37	0.48	0.6	0.71
0.002	0.25	0.28	0.37	0.48	0.6	0.71
0.005	0.25	0.28	0.37	0.48	0.6	0.71
0.01	0.25	0.28	0.37	0.48	0.6	0.71
0.02	0.25	0.28	0.36	0.48	0.6	0.7
0.05	0.24	0.27	0.36	0.47	0.59	0.7
0.1	0.24	0.27	0.35	0.47	0.58	0.69
0.2	0.25	0.27	0.35	0.46	0.57	0.68
0.5	0.33	0.35	0.4	0.48	0.57	0.66

Table C.3: Simulation results: MAD, DE prior, 'Coastguard' sequence

	σ_η					
c	0	1	2	3	4	5
0	0.36	0.4	0.44	0.48	0.51	0.54
0.001	0.36	0.4	0.45	0.48	0.51	0.54
0.002	0.36	0.4	0.45	0.48	0.51	0.54
0.005	0.36	0.4	0.45	0.48	0.52	0.54
0.01	0.36	0.4	0.45	0.48	0.51	0.54
0.02	0.36	0.4	0.45	0.48	0.51	0.54
0.05	0.36	0.4	0.44	0.48	0.51	0.54
0.1	0.37	0.4	0.44	0.48	0.51	0.54
0.2	0.4	0.41	0.45	0.48	0.51	0.54
0.5	0.47	0.46	0.48	0.49	0.51	0.54

Table C.4: Simulation results: MAD, DE prior, 'Mobile' sequence

Bibliography

[1] G. Baroni, G. Ferrigno, and R. Orecchia. Real-time three dimensional motion analysis for patient positioning verification. *Radiother Oncol*, 54:21–27, 2000.

[2] Y.-K. Chen. *True Motion Estimation - Theory, Application, and Implementation.* PhD thesis, Department of Electrical Engineering, Princeton University, 1998.

[3] C. Debes, C. Brown, S. Dueking, and A. Zoubir. Markov random fields for use in motion analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2006. under review.

[4] G. Foresti. Object recognition and tracking for remote video surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(7):1045–1062, 1999.

[5] B. Furht, J. Greenberg, and R. Westwater. *Motion Estimation algorithm for Video Compression.* Kluwer Academic Publishers, 1997.

[6] R. Gonzalez and R. Woods. *Digital Image Processing.* Prentice Hall, 2002.

[7] G. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 16(3):267–276, 1994.

[8] M. Hemmendorff. *Motion Estimation and Compensation in Medical Imaging.* PhD thesis, Department of Biomedical Engineering, Linköpings Universitet, 2001.

[9] B. Hunt and O. Kubler. Karhunen-Loève multispectral image restoration, Part I: Theory,. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 592–599, 1984.

[10] M. Joshi and S. Chaudhuri. Zoom based super-resolution through sar model fitting. In *Proceedings 2004 IEEE International Conference on Image Processing*, volume 3, pages 1775–1778, Singapore, 2004.

[11] M. Joshi, S. Chaudhuri, and P. Rajkira. Super-resolution imaging: Use of zoom as a cue. *Image & Vision Computing*, 22(14):1185–1196, 2004.

[12] B. Klaus and P. Horn. *Robot Vision*. MIT Press, McGraw-Hill, 1986.

[13] T. Komatsu, K. Aizawa, T. Igarashi, and T. Saito. Signal-processing based method for acquiring very high resolution image with multiple cameras and its theoretical analysis. In *Proc. Inst. Elec. Eng.*, volume 140, pages 19–25, 1993.

[14] O. Laligant, R. Leconge, F. Truchetet, O. Mahu, and A. Diou. Motion estimation for medical application in psychiatry. *Journal of Electronic Imaging*, 13(3):570–578, 2004.

[15] R. Molina, M. Vega, J. Abad, and A. Katsaggelos. Parameter estimation in Bayesian high-resolution image reconstruction with multisensors. *IEEE Transactions on Image Processing*, 12(12):1655–1667, 2003.

[16] M. Ng and N. Bose. Mathematical analysis of super-resolution methodology. *IEEE Signal Processing Magazine*, 20(3):62–74, May 2003.

[17] A. Oppenheim, R. Schafer, and J. Buck. *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[18] S. Park, M. Park, and M. Kang. Super-resolution image reconstruction: A technical overview. *IEEE Signal Processing Magazine*, 20(3):21–36, May 2003.

[19] I. Richardson. *H.264 and MPEG-4 Video Compression*. Wiley, 2003.

[20] B. Ripley. *Spatial Statistics*. Wiley, 1981.

[21] R. Schultz and R. Stevenson. Extraction of high-resolution frames from video sequences. *IEEE Transactions on Image Processing*, 5(6):996–1011, 1996.

[22] Y. Tsin, V. Ramesh, and T. Kanade. Statistical calibration of the CCD imaging process. In *Proceedings 2001 IEEE International Conference on Computer Vision*, volume 1, pages 480–487, Vancouver, BC, Canada, 2001.

[23] Z. Wang and A. Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, 2002.

[24] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[25] G. Winkler. *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods*. Springer, 2003.